ADVANCES IN CALIBRATION AND INTERPOLATION: CENSORED AND BIG DATA APPLICATIONS

A Dissertation
Presented to
The Academic Faculty

By

Fang Cao

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
H. Milton Stewart School of Industrial and Systems Engineering

Georgia Institute of Technology

August 2017

ADVANCES IN CALIBRATION AND INTERPOLATION: CENSORED AND BIG DATA APPLICATIONS

Approved by:

Roshan Joseph Vengazhiyil, Advisor
H. Milton Stewart School of Industrial and Systems Engineering
Georgia Institute of Technology

C. F. Jeff Wu
H. Milton Stewart School of Industrial and Systems Engineering
Georgia Institute of Technology

Jianjun Shi
H. Milton Stewart School of Industrial and Systems Engineering
Georgia Institute of Technology

Benjamin Haaland
School of Medicine
The University of Utah

William Brenneman
Procter & Gamble Company

Date Approved: July 17, 2017

Hiding within those mounds of data is knowledge that could change the life of a patient, or change the world.

Atul Butte

To my beloved parents and grandparents

for their support, encouragement and inspiration.

ACKNOWLEDGEMENTS

I would like to take the chance to express my gratitude to many people who have guided, inspired and supported me throughout my doctoral studies at Georgia Tech.

First, I would like to thank my advisor, Professor Roshan Joseph Vengazhiyil, for his tremendous effort in these years to help me become an independent and innovative researcher. Roshan is one of the top researchers in experimental design and modeling, and I can already learn a lot from his thoughtful and original works. Aside from that, he devotes all his heart to help me understand and learn his thinking process of conducting innovative research. Moreover, he is always thinking about what can benefit his students for their future life, not simply for completing near-term tasks. His guidance not only hones my research skills extensively, but has profound influence on my attitude to tackle problems in my life and career. It is a great honor and pleasure to work with Roshan, and I could not imaging being guided by a better advisor.

I would also like to thank Professor Jeff Wu, Professor Jianjun Shi, Professor Ben Haaland and Professor William Brenneman for serving on my dissertation committee. Professor Wu and Professor Shi are both leading figures who have shaped their respective domains. They shared with me their insights on my dissertation and pointed out the right direction for me. Professor Haaland, as a young and accomplished researcher, gave me very useful suggestions and inspired my enthusiasm for research. I am also grateful to Professor Brenneman for showing me the way to conduct influential research work in industry. I would like to extend my special thanks to Professor Kobi Abayomi and Professor Nagi Gebraeel for their guidance and support during my doctoral studies.

I am thankful to my mates and colleagues at Georgia Tech. The time spent with them makes the journey of doctoral studies much more joyful and memorable. Their talents greatly inspired me and let me not feel lonely on the road in search of knowledge. They offer me numerous helpful suggestions, and give me the courage to carry on during the

hardest times. It is my honor to study and work with such a group of kind and talented people.

Lastly, I would like to express my gratitude to my family. Without their support, care and inspiration, I cannot have the determination and courage to complete this journey. Finally, I would like to thank my girlfriend for her companion and support, and for sharing marvelous time with me.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Advances of computing capability and increasing demand for analyzing data from complex systems in various engineering fields have made computer experiments an inevitable tool for exploring and optimizing systems. Physical experiments are very costly to conduct in many applications such as a cardiovascular system study or a rocket engine design. With the aid of high-performance computing, the cost for expensive physical experiments can be reduced dramatically by running the simulation codes on computers. Due to the deterministic nature of computer codes, Gaussian process model or kriging is widely used for interpolation and calibration.

Chapter 1 of the thesis deals with model calibration using censored data. The purpose of model calibration is to use data from a physical experiment to adjust the computer model so that the predictions can become closer to reality. The classic Kennedy-O'Hagan approach is widely used for model calibration, which can account for the inadequacy of the computer model while simultaneously estimating the unknown calibration parameters. In many applications, the phenomenon of censoring occurs when the exact outcome of the physical experiment is not observed, but is only known to fall within a certain region. In such cases, the Kennedy-O'Hagan approach cannot be used directly, and we propose a method to incorporate the censoring information when performing model calibration. The method is applied to study the compression phenomenon of liquid inside a bottle. The results show significant improvement over the traditional calibration methods, especially when the number of censored observations is large.

Chapter 2 proposes an interpolation technique which can be used with large and unstructured data. Kriging is widely used for interpolation of unstructured data because of its ability to produce confidence intervals for predictions. The model is fitted to the data using maximum likelihood or cross validation-based methods. Unfortunately, the fitting is expense for large data because one evaluation of the objective function requires $O(n^3)$

operations, where n is the size of the data. There exist other interpolation techniques such as inverse distance weighting (IDW), which doesnt require any estimation and therefore can be easily used with large data. However, the performance of IDW can be significantly worse than kriging. In this chapter, we propose a kriging method that does not require any estimation from data and whose performance is much better than that of IDW. We also propose a novel approach to choose nuggets in kriging that can produce numerically stable results, which is important for applying the technique to unstructured data. A technique for adaptively choosing the kernels is also developed.

Chapter 3 extends the automatic kriging proposed in Chapter 2 by exploiting the sequential nature of the adaptive modeling method. When more computing resource is available, we have the option to make estimates from adaptive nugget and adaptive kernel more accurate. A two-stage version of adaptive nugget predictor is proposed which is shown to outperform the state-of-the-art methods in terms of prediction accuracy. We also propose fast estimation techniques to improve the adaptive kernel predictor. The improved predictor is demonstrated to have enhanced stability and predictive performance over the traditional kriging method according to various simulation studies.

CHAPTER 1
MODEL CALIBRATION WITH CENSORED DATA

The purpose of model calibration is to make the model predictions closer to reality. The classical Kennedy-O'Hagan approach is widely used for model calibration, which can account for the inadequacy of the computer model while simultaneously estimating the unknown calibration parameters. In many applications, the phenomenon of censoring occurs when the exact outcome of the physical experiment is not observed, but is only known to fall within a certain region. In such cases, the Kennedy-O'Hagan approach cannot be used directly, and we propose a method to incorporate the censoring information when performing model calibration. The method is applied to study the compression phenomenon of liquid inside a bottle. The results show significant improvement over the traditional calibration methods, especially when the number of censored observations is large.

1.1 Introduction

Computer models are developed based on several simplifying assumptions of the physical system for the reason of mathematical tractability and therefore, the predictions based on computer models can go wrong when the assumptions are violated. The models can also contain unknown parameters known as calibration parameters, which need to be specified before a prediction can be made. Misspecification of these parameters can also lead to wrong predictions. Thus, to make the predictions meaningful and closer to reality, the calibration parameters need to be estimated accurately to adjust for possible model bias. In a fundamental work [1], the authors proposed a Gaussian process-based Bayesian framework for doing this. Follow-up works on this important problem of model calibration include [2], [3], [4], [5], [6], [7], [8], and [9], among many others.

In some applications, the exact value of the outcome of an experiment can be unknown

and only observed to be within a certain range. Such phenomenon of censorship is common in medical research and reliability studies. For example, in reliability testing of the components in a system, the engineer may terminate all the tests after a specific time point. In this case, we know the time-to-failure of the components to be greater than a specific value but do not observe the exact time. There is an extensive literature in dealing with censored data in survival analysis and reliability, see [10], [11], [12], among others. In the model calibration context, it is also possible to have censored observations. For instance, this happens when the response value is out of range of a measuring instrument. As related work, [13], [14] studies kriging with inequality-type data and data with qualitative information like continuity. [15] studies Gaussian process model with shape constraints such as monotonicity and convexity. To the best of the author's knowledge, there is no existing work that studies the calibration problem in the presence of censored data, and the purpose of this chapter is to fill in this gap and develop a methodology for such applications.

The chapter is organized as follows: The motivating application on liquid stability forecasting is described in Section 1.2. The model calibration problem with the presence of censored data is formalized in Section 1.3, where the parameter estimation as well as prediction for a future observation are provided under a Bayesian framework. In section 1.4 we introduce an approximate method which greatly reduces the computational effort. Section 1.5 shows the results of applying the proposed calibration approach to both the simulated and the real data. Concluding remarks are given in Section 1.6.

1.2 Motivating Application: Liquid Stability Forecasting

The motivation of the study is to predict the separation rate of a liquid product that has been fully mixed at the time of manufacture but can separate over time resulting in a clear liquid at the top. Separation is undesirable as some of the ingredients needed for full product benefit are not found in the clear liquid resulting in poor product performance. Therefore it is desirable to be able to predict the amount of separation early on in the design

of a new product in order to stay away from parts of the design space that lead to product separation. Having a first principles model to predict the stability of a given product can reduce or eliminate costly and time consuming physical stability studies.

In our case, we have a simple computer model based on the underlying physics of the separation process that provides a closed form equation for the rate of separation (v) as a function of several measurable factors. The measurable factors include the viscosity of the clear liquid (x_1), the density of the clear liquid (x_2), and the volume fraction of four key components ($\phi_1, \phi_2, \phi_3, \phi_4$). The physics model is given by

$$v = \frac{\lambda k_0 g \sum_{i=1}^{4} (\rho_i - x_2)\phi_i}{x_1},$$

where $k_0 = ((1-p)a_1 + pa_2)^2(\phi_1 + \phi_2)^{-2/(3-d_f)}$, and $p = \phi_2/b_2/(\phi_2/b_2 + \phi_1/b_1)$. The general form of this physics model can be found in [16].

In these equations, g is the gravitational constant, $\rho_1, ..., \rho_4, a_1, a_2, b_1, b_2$ are other known constants, λ is the calibration parameter and d_f is the fractal dimension that is within the range 1.8 to 2.2. The unit for velocity of the separation (v) is in mm/day. The scientists are quite confident about the sign of v, so the interests center around only the $|v|$. Furthermore, we used the logarithm of $|v|$ as the response for variance stabilization. Thus, the computer model for $y = \log|v|$ is

$$y = \log\left|\frac{\lambda k_0 g \sum_{i=1}^{4} (\rho_i - x_2)\phi_i}{x_1}\right|.$$

We performed a designed experiment that spanned the ranges of x_1, x_2, $\phi_1, ..., \phi_4$ in an attempt to cover a large space to validate and tune the model. The observed separation rate is measured by filling glass bottles with the product and then placing them in controlled storage rooms that have cameras outfitted to take pictures every hour. In Figure 1.1, we show a set of pictures for two samples placed in the control room for 240 days where we show the initial picture, the picture at 120 days, and the picture at 240 days. Note that the first set of pictures does not show any separation that is measurable and is thus below the

limit of detection or threshold. The second set of pictures shows the separation rate that is large enough to be measurable. We used a software that reads in the pixel images to track the separation over time. The computer program automatically fits a regression line and outputs the separation rate v (mm/day).

We used a 204-run D-optimal design based on a quadratic response surface model to conduct the experiment. Out of the 204 measurements, 109 showed so little separation that the estimated regression slopes were not statistically significant and therefore, the computer program did not produce any values of v. Thus we have 109 left-censored observations in which all we know is that $|v|$ is less than the threshold t given by 1.32×10^{-2} mm/day. We know that if the samples were placed in the control room for a longer period of time, the separation rates of all the samples would ultimately be measured. But project teams typically need to move fast requiring analysis of the data as soon as possible, including the analysis of the censored data. In the next section we develop a general methodology for model calibration that can handle censored data.

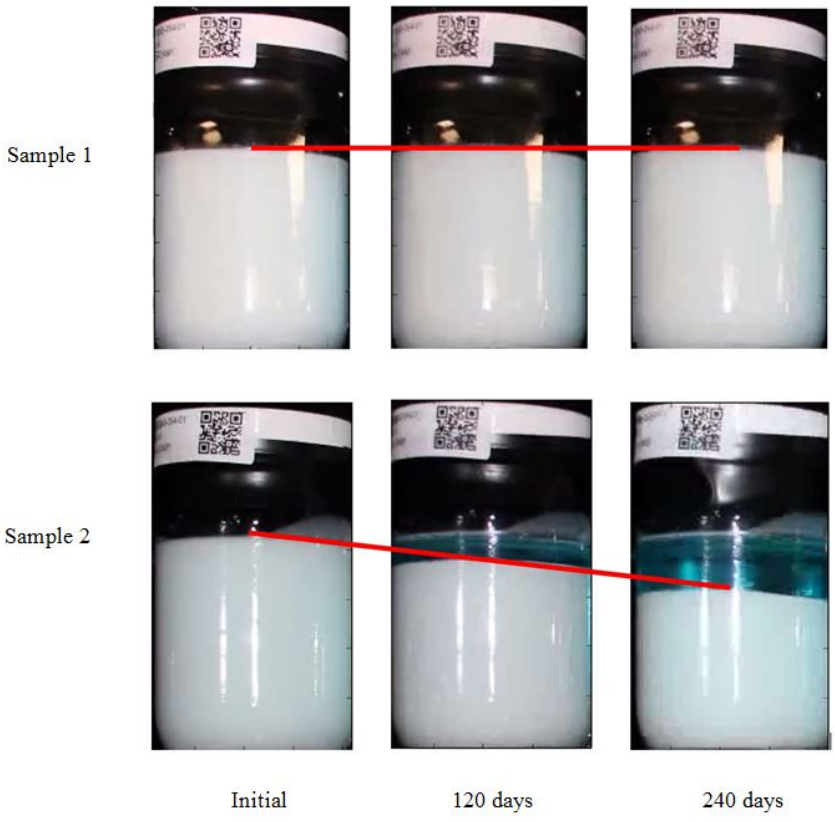

Figure 1.1: Two samples with pictures taken at initial placement, 120 days and 240 days

1.3 Methodology

Suppose we have n physical experiments in total. Let $\boldsymbol{y}$ denote the vector of responses for the n runs. We also have the computer code to simulate the output. The input of the computer code is composed of two parts: the control variables $\boldsymbol{x} = (x_1, \ldots, x_d)^T$ and the calibration parameters $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_q)^T$. The calibration parameters represent some intrinsic properties of the physical system and are unknown. With each set of input variable $(\boldsymbol{x}, \boldsymbol{\lambda})$, the computer code gives a deterministic output $f(\boldsymbol{x}, \boldsymbol{\lambda})$. For the ease of formulation, assume that the functional form of the computer code is known. In other words, we assume that the computer code can be executed at virtually no cost. As a matter of fact, if we have an "expensive" computer code which results in a long running time, the modeling framework is essentially the same, except that the computer code is also modeled using the

Gaussian process framework. [1] formalizes the calibration problem in the following way

$$y_i = f(\boldsymbol{x}_i, \boldsymbol{\lambda}) + \delta(\boldsymbol{x}_i) + \epsilon_i, \tag{1.1}$$

where $\delta(\cdot)$ is the unknown discrepancy function which accounts for the model inadequacy, and $\epsilon_i \overset{i.i.d.}{\sim} N(0, \sigma^2)$ is the observational error, $i = 1, ..., n$. The discrepancy function $\delta(\cdot)$ is modeled as a realization of a Gaussian process with zero mean, constant variance τ^2 and a correlation function $R(\cdot, \cdot)$. The most commonly used correlation function is the Gaussian correlation given by

$$R(\boldsymbol{x}, \boldsymbol{x'}) = \exp\{-\sum_{j=1}^{d} \theta_j (x_j - x'_j)^2\},$$

where $\boldsymbol{\theta} = (\theta_1, \theta_2, ..., \theta_d)^T$ are called correlation parameters. Let $\boldsymbol{\psi} = (\boldsymbol{\theta}^T, \tau^2, \sigma^2)^T$ denote the vector of hyper-parameters in the model. The Kennedy-O'Hagan approach utilizes a Bayesian framework to estimate $\boldsymbol{\lambda}, \boldsymbol{\psi}$ and conduct the prediction for new observations.

1.3.1 Model Calibration with Censored Data

Suppose we do not observe the exact values of some outcomes but their values are known to fall within a certain region. Let the response vector $\boldsymbol{y}$ be composed of two parts $\boldsymbol{y}^T = (\boldsymbol{y}_o^T, \boldsymbol{y}_c^T)$ where the n_o-dimensional vector $\boldsymbol{y}_o$ denotes the outcome of the observed data, and the n_c-dimensional vector $\boldsymbol{y}_c$ denotes the outcome of the censored data. Note that the actual values of $\boldsymbol{y}_c$ are not observed, and we only know that $\boldsymbol{y}_c$ falls within certain censoring region C. For the moment assume that C is a hyper-rectangle, namely, $C = [a_1, b_1] \times [a_2, b_2] \times ... \times [a_{n_c}, b_{n_c}]$. Let $\boldsymbol{X} = (\boldsymbol{X}_o^T, \boldsymbol{X}_c^T)^T$ denote the matrix of control variables with rows $\boldsymbol{x}_1^T, \boldsymbol{x}_2^T, ..., \boldsymbol{x}_n^T$. According to the model in (1.1), $\boldsymbol{y}|\boldsymbol{\lambda}, \boldsymbol{\psi} \sim N(\boldsymbol{f}(\boldsymbol{\lambda}), \boldsymbol{\Sigma})$ with $\boldsymbol{f}(\boldsymbol{\lambda}) = (\boldsymbol{f}_o^T(\boldsymbol{\lambda}), \boldsymbol{f}_c^T(\boldsymbol{\lambda}))^T$ and

$$\boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_o & \boldsymbol{\Sigma}_{oc} \\ \boldsymbol{\Sigma}_{oc}^T & \boldsymbol{\Sigma}_c \end{bmatrix} = \tau^2 \begin{bmatrix} \boldsymbol{R}_o + \rho \boldsymbol{I}_o & \boldsymbol{R}_{oc} \\ \boldsymbol{R}_{oc}^T & \boldsymbol{R}_c + \rho \boldsymbol{I}_c \end{bmatrix},$$

where $\boldsymbol{R}_o$ is the $n_o \times n_o$ matrix with jkth element $R(\boldsymbol{X}_o^{(j)}, \boldsymbol{X}_o^{(k)})$, $\boldsymbol{X}_o^{(j)}$ is the jth row of $\boldsymbol{X}_o$, $\boldsymbol{R}_{oc}$ is the $n_o \times n_c$ matrix with jkth element $R(\boldsymbol{X}_o^{(j)}, \boldsymbol{X}_c^{(k)})$, $\rho = \sigma^2/\tau^2$, and $\boldsymbol{I}_o$ is the $n_o \times n_o$ identity matrix. The matrices $\boldsymbol{R}_c$ and $\boldsymbol{I}_c$ are defined similarly.

Let $\mathcal{D}$ denote the data $\{\boldsymbol{y}_o, \boldsymbol{y}_c \in C\}$. The likelihood function is then given by

$$\begin{aligned} L(\boldsymbol{\lambda}, \boldsymbol{\psi}; \mathcal{D}) &= \int_C p(\boldsymbol{y}_o, \boldsymbol{y}_c | \boldsymbol{\lambda}, \boldsymbol{\psi}) d\boldsymbol{y}_c \\ &= p(\boldsymbol{y}_o | \boldsymbol{\lambda}, \boldsymbol{\psi}) \int_C p(\boldsymbol{y}_c | \boldsymbol{y}_o, \boldsymbol{\lambda}, \boldsymbol{\psi}) d\boldsymbol{y}_c, \end{aligned} \tag{1.2}$$

where

$$\boldsymbol{y}_o | \boldsymbol{\lambda}, \boldsymbol{\psi} \sim N(\boldsymbol{f}_o(\boldsymbol{\lambda}), \boldsymbol{\Sigma}_o) \text{ and} \tag{1.3}$$

$$\boldsymbol{y}_c | \boldsymbol{y}_o, \boldsymbol{\lambda}, \boldsymbol{\psi} \sim N(\boldsymbol{f}_c(\boldsymbol{\lambda}) + \boldsymbol{\Sigma}_{oc}^T \boldsymbol{\Sigma}_o^{-1} (\boldsymbol{y}_o - \boldsymbol{f}_o(\boldsymbol{\lambda})), \boldsymbol{\Sigma}_c - \boldsymbol{\Sigma}_{oc}^T \boldsymbol{\Sigma}_o^{-1} \boldsymbol{\Sigma}_{oc}). \tag{1.4}$$

[17] has studied Gaussian process regression with censored data. They use a latent variable approach, which makes the dimensionality of the integral in their likelihood equal to the total number of observations. The integrand, also known as Tobit likelihood [18], cannot be integrated analytically making the likelihood computationally intractable. They used Expectation Propagation method [19] to approximate the likelihood and obtain the posterior distribution. On the other hand, the likelihood in our formulation given in (1.2) is much simpler with dimensionality equal to the number of censored observations, which can be much smaller than the total number of observations. Moreover, we can take advantage of the multivariate normal form of the integrand and calculate the exact value of the likelihood. Specifically, we use an algorithm for calculating multivariate normal probabilities developed by [20] which is implemented in the R package *mvtnorm* [21].

Suppose the prior distribution for the calibration parameters is independent of the hyperparameters $\boldsymbol{\psi}$, i.e. $\pi(\boldsymbol{\lambda}, \boldsymbol{\psi}) = \pi(\boldsymbol{\lambda})\pi(\boldsymbol{\psi})$. Then the posterior distribution of the parameters

is given by

$$p(\boldsymbol{\lambda}, \boldsymbol{\psi}|\mathcal{D}) \propto \pi(\boldsymbol{\lambda})\pi(\boldsymbol{\psi})L(\boldsymbol{\lambda}, \boldsymbol{\psi}; \mathcal{D}), \tag{1.5}$$

upon which we can make Bayesian inference for the set of parameters. Let $(\hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\psi}})$ denote the joint *maximum-a-posteriori* estimate of $(\boldsymbol{\lambda}, \boldsymbol{\psi})$. As in [1], we have chosen to fix $\boldsymbol{\psi}$ at $\hat{\boldsymbol{\psi}}$ which makes the analysis computationally tractable.

1.3.2 Prediction of Future Observation

In real world applications, it is often of interest to predict the response (y_{new}) at a new location $\boldsymbol{x}_{new}$, which is also a major purpose of model calibration. Given the data, the predictive distribution for y_{new} can be expressed as

$$p(y_{new}|\mathcal{D}) = \int \Big\{ \frac{\int_C p(\boldsymbol{y}_o, \boldsymbol{y}_c|\boldsymbol{\lambda}) p(y_{new}|\boldsymbol{y}_o, \boldsymbol{y}_c, \boldsymbol{\lambda}) d\boldsymbol{y}_c}{\int_C p(\boldsymbol{y}_o, \boldsymbol{y}_c|\boldsymbol{\lambda}) d\boldsymbol{y}_c} \Big\} p(\boldsymbol{\lambda}|\mathcal{D}) d\boldsymbol{\lambda}, \tag{1.6}$$

where

$$p(\boldsymbol{\lambda}|\mathcal{D}) = \frac{\pi(\boldsymbol{\lambda})L(\boldsymbol{\lambda}, \hat{\boldsymbol{\psi}}; \mathcal{D})}{\int \pi(\boldsymbol{\lambda})L(\boldsymbol{\lambda}, \hat{\boldsymbol{\psi}}; \mathcal{D}) d\boldsymbol{\lambda}}. \tag{1.7}$$

It is easy to show that

$$y_{new}|\boldsymbol{y}_o, \boldsymbol{y}_c, \boldsymbol{\lambda} \sim N(\mu_{y_{new}}, \sigma^2_{y_{new}}), \tag{1.8}$$

where

$$\mu_{y_{new}} = f(\boldsymbol{x}_{new}, \boldsymbol{\lambda}) + \hat{\tau}^2 \boldsymbol{r}^T(\boldsymbol{x}_{new}) \boldsymbol{\Sigma}^{-1} \begin{bmatrix} \boldsymbol{y}_o - \boldsymbol{f}_o(\boldsymbol{\lambda}) \\ \boldsymbol{y}_c - \boldsymbol{f}_c(\boldsymbol{\lambda}) \end{bmatrix}, \tag{1.9}$$

$$\sigma^2_{y_{new}} = \hat{\tau}^2 + \hat{\sigma}^2 - \hat{\tau}^4 \boldsymbol{r}^T(\boldsymbol{x}_{new}) \boldsymbol{\Sigma}^{-1} \boldsymbol{r}(\boldsymbol{x}_{new}). \tag{1.10}$$

Here, $\boldsymbol{r}(\boldsymbol{x}_{new})$ is the n-dimensional column vector with jth element $R(\boldsymbol{x}_{new}, \boldsymbol{x}_j), j = 1, ..., n$. Now to obtain $p(y_{new}|\mathcal{D})$, we can sample as follows. First, we can use a Metropolis-Hastings algorithm to sample $\boldsymbol{\lambda}$ from (1.7), then sample $\boldsymbol{y}_c$ from the multivariate normal distribution in (1.4) truncated to $\boldsymbol{y}_c \in C$, and finally sample y_{new} from (1.8). This pro-

cedure is computationally intensive because sampling from multivariate truncated normal distribution can be prohibitive when n_c is large [22]. The procedure becomes even more difficult to use when we need to predict the observations at numerous locations of $\boldsymbol{x}$, although some low-rank approximation methods can be used to simplify this task [23]. If we can obtain explicit expressions for the mean and variance of $y_{new}|\mathcal{D}$, we can then approximate the predictive distribution using normal distribution and easily obtain the prediction and the corresponding confidence interval. Another approximation approach that allows for exact Bayesian inference will be presented in the next section.

As shown in Appendix A1, the expectation of the predictive distribution of y_{new} is given by

$$E(y_{new}|\mathcal{D}) = E_{\boldsymbol{\lambda}|\mathcal{D}} f(\boldsymbol{x}_{new}, \boldsymbol{\lambda}) + \hat{\tau}^2 \boldsymbol{r}^T(\boldsymbol{x}_{new}) \boldsymbol{\Sigma}^{-1} \begin{bmatrix} \boldsymbol{y}_o - E_{\boldsymbol{\lambda}|\mathcal{D}} \boldsymbol{f}_o(\boldsymbol{\lambda}) \\ E_{\boldsymbol{\lambda}|\mathcal{D}} \{\tilde{\boldsymbol{f}}_c(\boldsymbol{\lambda}) - \boldsymbol{f}_c(\boldsymbol{\lambda})\} \end{bmatrix}, \quad (1.11)$$

where $\tilde{\boldsymbol{f}}_c(\boldsymbol{\lambda}) = E(\boldsymbol{y}_c|\mathcal{D}, \boldsymbol{\lambda})$. This can serve as an easy-to-evaluate surrogate model for the output. To compute (1.11), first we need to evaluate the expectation of a multivariate truncated normal distribution, $E(\boldsymbol{y}_c|\mathcal{D}, \boldsymbol{\lambda})$. [24] derived the first and second moments of multivariate truncated normal distribution for the case of left truncation, i.e., $b_i = +\infty, i = 1, 2, ..., n_c$ for the hyper-rectangular region C. [25] and [26] extended the results and were able to compute higher moments for truncation in a general hyper-rectangular region. Since only the first and second moments are used for approximating the predictive distribution, we still adopt Tallis' method which provides the explicit expressions of the moments and the derivation for the general hyper-rectangular region is shown in Appendices A2 and A4. According to (1.16) in Appendix A2, computing expectation of multivariate truncated normal requires n_c evaluations of $(n_c - 1)$-dimensional multivariate normal cumulative distribution function. Joint use of (1.11) and (1.16) enables us to compute the expectation of the predictive distribution.

The variance of the predictive distribution can be obtained as (see Appendix A3)

$$var(y_{new}|\mathcal{D}) = var_{\boldsymbol{\lambda}|\mathcal{D}}\left\{ f(\boldsymbol{x}_{new},\boldsymbol{\lambda}) + \hat{\tau}^2\boldsymbol{r}^T(\boldsymbol{x}_{new})\boldsymbol{\Sigma}^{-1}\left[\boldsymbol{y}_o^T - \boldsymbol{f}_o^T(\boldsymbol{\lambda}), \tilde{\boldsymbol{f}}_c^T(\boldsymbol{\lambda}) - \boldsymbol{f}_c^T(\boldsymbol{\lambda})\right]^T \right\}$$
$$+ \boldsymbol{t}_{new}^T E_{\boldsymbol{\lambda}|\mathcal{D}}\tilde{\boldsymbol{\Sigma}}_c(\boldsymbol{\lambda})\boldsymbol{t}_{new} - \hat{\tau}^4\boldsymbol{r}^T(\boldsymbol{x}_{new})\boldsymbol{\Sigma}^{-1}\boldsymbol{r}(\boldsymbol{x}_{new}) + \hat{\tau}^2 + \hat{\sigma}^2, \qquad (1.12)$$

where

$$\boldsymbol{t}_{new}^T = \hat{\tau}^2\boldsymbol{r}^T(\boldsymbol{x}_{new})\boldsymbol{\Sigma}^{-1}_{[,n_o+1:n]}.$$

Here $\boldsymbol{\Sigma}^{-1}_{[,n_o+1:n]}$ denotes the last n_c columns of $\boldsymbol{\Sigma}^{-1}$ and $\tilde{\boldsymbol{\Sigma}}_c(\boldsymbol{\lambda}) = var(\boldsymbol{y}_c|\mathcal{D},\boldsymbol{\lambda})$ is the conditional covariance matrix of a multivariate truncated normal distribution. According to (1.18) in Appendix A4, computing the covariance matrix involves about $n_c^2/2$ evaluations of (n_c-2)-dimensional multivariate normal cumulative distribution function. Moreover, these computations need to be repeated for each posterior sample of $\boldsymbol{\lambda}$. Thus the total computational effort can be large even with the normal approximation of the predictive distribution.

1.4 Approximate Method

In the previous section we have proposed an exact method for model calibration with censored data. Although we are able to compute the likelihood exactly through some efficient algorithms, the inference and prediction can still be time consuming when the number of censored observations is large. Therefore, it is worthwhile to investigate approximate methods for efficient computation.

Suppose an observation y is censored in an interval $[a,b]$. The simplest method to avoid dealing with the censored data is to replace them with the midpoint of the interval $(a+b)/2$. This only works well if the interval $[a,b]$ is narrow, but not when it is wide. Moreover, such a method ignores the uncertainties in the censored data, that is, the actual value could be anywhere in the interval $[a,b]$. In this section we propose a simple but effective method to

incorporate those uncertainties.

Consider the problem of approximating a uniform distribution $U(a,b)$ with a normal distribution $N(\mu,\nu)$. The Hellinger distance between the two densities is given by

$$\int_{\mathbb{R}} \left\{ \phi^{1/2}(y;\mu,\nu) - \frac{I_{(a,b)}(y)}{\sqrt{b-a}} \right\}^2 dy,$$

which can be minimized with respect to μ and ν. Thus,

$$\begin{aligned}
(\mu^*,\nu^*) &= \arg\min_{\mu,\nu} \int_{\mathbb{R}} \left\{ \phi^{1/2}(y;\mu,\nu) - \frac{I_{(a,b)}(y)}{\sqrt{b-a}} \right\}^2 dy \\
&= \arg\max_{\mu,\nu} \int_a^b \phi^{1/2}(y;\mu,\nu) dy \\
&= \arg\max_{\mu,\nu} \nu^{1/4} \left\{ \Phi\left(\frac{b-\mu}{\sqrt{2\nu}}\right) - \Phi\left(\frac{a-\mu}{\sqrt{2\nu}}\right) \right\} \\
&= \left(\frac{a+b}{2}, 0.0638(b-a)^2 \right).
\end{aligned}$$

The approximation is shown in the left panel of Figure 1.2 for $U(0,1)$. Using this normal approximation, we can easily accommodate the censored data into the classic calibration framework. For censored observations we just add one more term to (1.1) such that

$$y = f(\boldsymbol{x},\boldsymbol{\lambda}) + \delta(\boldsymbol{x}) + e + \epsilon, \tag{1.13}$$

where $y = \mu^*$ is treated as the observed value and $e \sim N(0,\nu^*)$ represents the extra uncertainty due to the censorship. Thus, essentially we are using the same model as in (1.1) with the censored observations replaced by μ^* but with a larger variance $\sigma^2 + \nu^*$. The larger variance accounts for the additional uncertainty in the data introduced due to censoring. The simplicity of this approach is that all the existing techniques for estimation and prediction in model calibration can be used without the need of any specialized tools to deal with the censored data.

The model is postulated on $\log|v|$ for our liquid stability forecasting application, so we

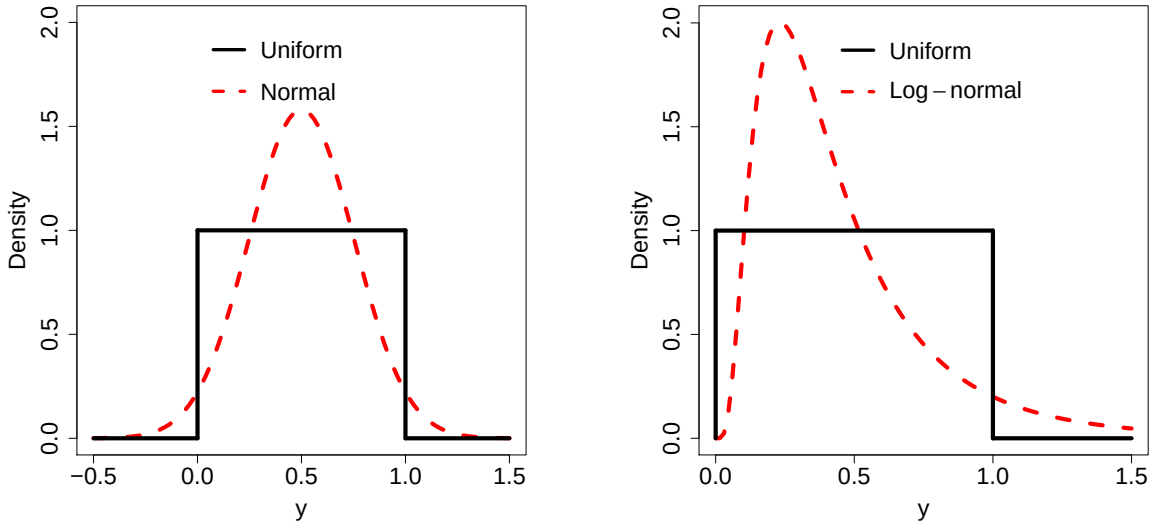

Figure 1.2: Normal (left) and log-normal (right) approximations of $U(0,1)$.

need some modifications. More specifically, now we aim to approximate a $U(0,t)$ with a log-normal distribution $LN(\mu,\nu)$. Then,

$$\begin{aligned}(\mu^*,\nu^*) &= \arg\min_{\mu,\nu}\int_{\mathbb{R}}\left\{\frac{\exp\{-(\log y-\mu)^2/4\nu\}}{(2\pi\nu y^2)^{1/4}}-\frac{I_{(0,t)}(y)}{\sqrt{t}}\right\}^2 dy\\ &= \arg\max_{\mu,\nu}\int_0^t \frac{\exp\{-(\log y-\mu)^2/4\nu\}}{(2\pi\nu y^2)^{1/4}}dy\\ &= \arg\max_{\mu,\nu}\nu^{1/4}\exp\left(\frac{2\mu+\nu}{4}\right)\Phi\left(\frac{\log t-\mu-\nu}{\sqrt{2\nu}}\right)\\ &= (\log t-1, 0.4676).\end{aligned}$$

The right panel of Figure 1.2 displays the approximation using log-normal distribution for $U(0,1)$. It does not look like a great approximation, but definitely should work better than a point mass distribution at $t/2$. The approximation can be further improved using a mixture of normal distributions as in [27]. The details are given in Appendix A5.

1.5 Results

In this section, we illustrate the performance of the proposed calibration methods by first using a simulated example and then using the motivating example on liquid stability forecasting. Three methods are used in the comparison: the original Kennedy-O'Hagan (KO) approach using only the observed data, the exact calibration method developed in Section 3, and the approximate calibration method using the normal approximation proposed in Section 4.

1.5.1 Simulation Example

Suppose the underlying true function is $y = 2x + 0.5 - 4(x - 0.5)^2$ where we use $y = \lambda x + 0.5$ as the computer model and $\delta(x) = -4(x - 0.5)^2$ represents the model bias. For observed data, we sampled six design points $\boldsymbol{x}_o$ uniformly from $[0, 0.043] \cup [0.138, 1]$ with two replicates. The data are generated using $y(x_o) = 2x_o + 0.5 - 4(x_o - 0.5)^2 + \epsilon$, where $\epsilon \sim N(0, 0.03)$. We also sampled ten design points $\boldsymbol{x}_c$ uniformly from $[0.043, 0.138]$ for censored data. Note that $y(x_c) \in [-0.25, 0.25]$ is taken as the censoring interval.

We applied the three aforementioned calibration methods (exact, approximate and observed) to the data where $\epsilon \sim N(0, \sigma^2)$ and $\delta(x)$ follows a Gaussian process with mean zero, variance τ^2, and correlation function $R(x, x') = \exp\{-\theta(x - x')^2\}$. We used non-informative priors for the parameters τ^2 and σ^2, a uniform prior $U(1, 3)$ for λ and $U(0, 150)$ for θ. We repeated these simulations 1000 times (see Figure 1.3 one specific case) and compared the mean squared prediction error MSPE $= \sum_{i=1}^{n}(y_i - \hat{y}_i)^2/n$, the estimated calibration parameter $\hat{\lambda}$, and the coverage rate η given by

$$\eta = \frac{1}{n}\#\{y_i \in [\hat{L}_i, \hat{U}_i], i = 1, ..., n\},$$

where $\hat{y}_i$ is the predicted response, $\hat{L}_i$ and $\hat{U}_i$ are the lower and upper bounds of the 95% prediction interval. The prediction intervals for the approximate and observed methods are

computed based on the quantiles of predictions from the MCMC samples. For the exact method, the bounds are approximated by $\hat{L}_i = \hat{y}_i - z_{0.975}\sqrt{V_i}, \hat{U}_i = \hat{y}_i + z_{0.975}\sqrt{V_i}$ and V_i is the predictive variance. The results are plotted in Figure 1.4. We can see that both the exact and the approximate methods clearly outperform the KO approach that ignores the censored observations. Note that all the three methods are biased in estimating the calibration parameter λ due to the identifiability issue of the Kennedy-O'Hagan calibration framework [6, 28].

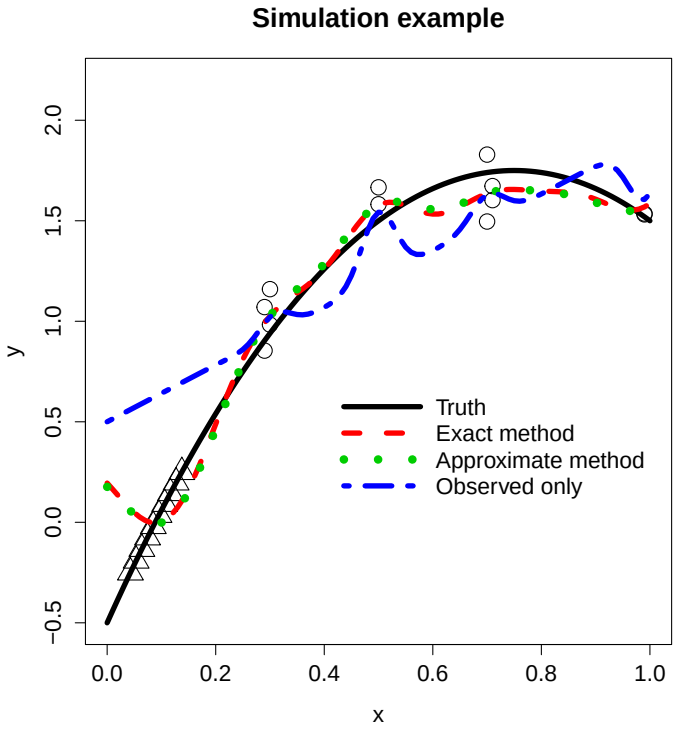

Figure 1.3: A specific case of the simulated example (circle represents observed data and triangle represents censored data).

1.5.2 Application to Liquid Stability Forecasting

We now return to the study of separation rate in the process of liquid compression described in Section 1.2. As shown in Figure 1.5, a log-normal prior is postulated for λ based on previous experiments (the details are omitted due to confidentiality reasons). We specify inverse gamma prior $IG(1, 2)$ for the variance parameter τ^2 based on former

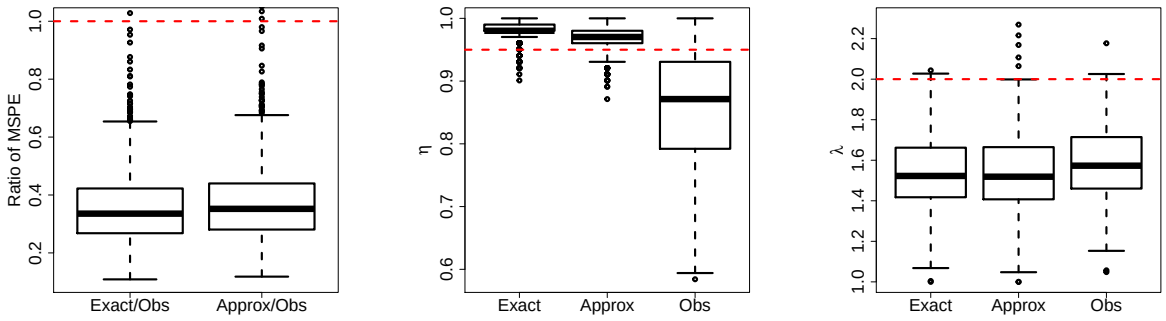

Figure 1.4: Comparison of the three calibration methods using the simulated example (MSPE is smaller-the-better, η should be close to 0.95, and λ should be close to 2).

experience. We have replicates in our experimental design so that error variance σ_i^2 for each individual observation can be estimated beforehand, $i = 1, ..., n$. It is not easy to postulate a prior for the correlation parameter $\boldsymbol{\theta}$. One possible choice proposed by [29] is $\theta_i \overset{i.i.d.}{\sim} Gamma(2, \bar{d}^2/\log(2))$ for $i = 1, ..., 6$, where $\bar{d}^2$ is the harmonic mean of the pairwise squared distance between the design points. We used the *maximum-a-posteriori* estimate of $\boldsymbol{\psi}$ based on (1.5) as the plug-in estimator for prediction. Since the posterior distribution of λ is concentrated around its mode according to Figure 1.5, we also fixed λ at its *maximum-a-posteriori* estimate to simplify the computations. The performance of the methods is assessed from two aspects. For the observed data, the mean squared prediction error (MSPE) and coverage rate η are computed as in the previous section:

$$\text{MSPE} = \frac{1}{n_o} \sum_{y_i > t', i=1,...,n} (y_i - \hat{y}_i)^2,$$
$$\eta = \frac{1}{n_o} \#\{y_i > t', y_i \in [\hat{L}_i, \hat{U}_i], i = 1, ..., n\},$$

where $t' = \log(t)$ is the censoring threshold for y, $\hat{y}_i$ is the predicted response, $\hat{L}_i$ and $\hat{U}_i$ are the lower and upper bounds of the 95% prediction interval. For censored data, we cannot use these two measures because the exact separation rate is unknown. Instead, the following three measures: false positive rate α, lower-bound coverage rate γ and upper-

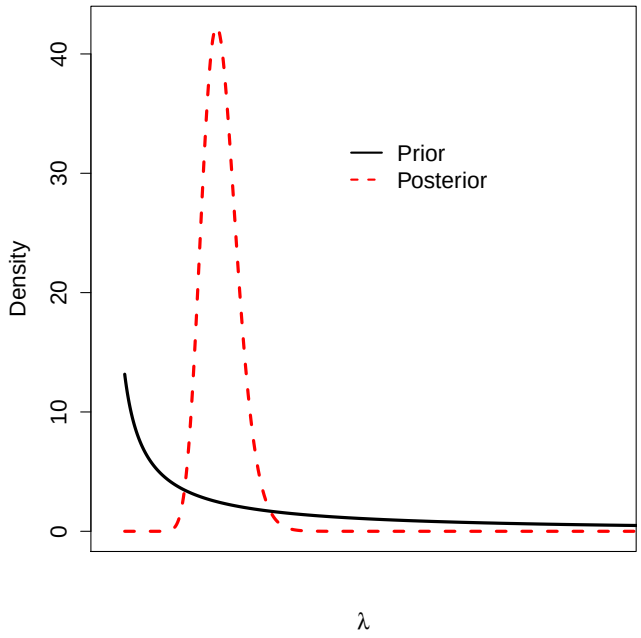

Figure 1.5: Prior and posterior distribution of λ in the liquid stability forecasting application.

bound coverage rate ζ are adopted, which are defined as

$$\alpha = \frac{1}{n_c}\#\{y_i \leq t', \hat{y}_i > t', i = 1, ..., n\},$$
$$\gamma = \frac{1}{n_c}\#\{y_i \leq t', \hat{L}_i \leq t', i = 1, ..., n\},$$
$$\zeta = \frac{1}{n_c}\#\{y_i \leq t', \hat{U}_i \leq t', i = 1, ..., n\}.$$

We used a five-fold cross-validation procedure to compare the performance so that the results are less prone to over-fitting. The summary of the performance measures is displayed in Table 1.1.

We find that the approximate method achieves the best MSPE and coverage rate η for observed data. On the other hand, exact method's performance for censored observations is the best according to α, γ, and ζ. In general, the exact and the approximate methods both have better overall performance than the calibration method using only the observed

Table 1.1: The comparison of the performance of model calibration methods (MSPE, α smaller the better, η should be close to 0.95, γ, ζ larger the better)

	Observed	Approximate	Exact
MSPE	0.182	**0.139**	0.154
η	0.937	**0.947**	0.937
α	0.129	0.129	**0.083**
γ	0.991	0.991	**1**
ζ	0.230	0.403	**0.596**

data. As expected, the approximate method is much faster than the exact method. It took only 1.3 hours for the approximate method for the computations within each fold, whereas it took 40.4 hours for the exact method in a 3.33 GHz desktop.

1.6 Conclusions

This chapter proposes an approach for solving model calibration problem when censored observations are present in the physical experiment data. Following Kennedy and O'Hagan's approach, a Gaussian process framework is adopted for the estimation of calibration parameters and model bias. Compared to the existing literature dealing with censored data in Gaussian process regression, our proposed calibration method uses exact computation of the likelihood. We also proposed a computationally efficient approximate method, whose performance is found to be only slightly inferior to that of the exact method. A great advantage of the approximate method is that all the existing methods for model calibration can be used to deal with the censored data with only minor modifications. Therefore we recommend using the approximate method when the number of censored observations is large or the censoring interval is narrow, and otherwise the exact method should be preferred. The approximate method can be further improved using a mixture normal approximation as discussed in Appendix A5.

1.7 Appendix

A1. Expectation of the predictive distribution

The expectation of the predictive distribution given $\boldsymbol{\lambda}$ can be expressed as

$$E(y_{new}|\mathcal{D},\boldsymbol{\lambda}) = \frac{\int_{\mathbb{R}}\int_C y_{new}p(y_{new},\boldsymbol{y}_o,\boldsymbol{y}_c|\boldsymbol{\lambda})dy_{new}d\boldsymbol{y}_c}{\int_C p(\boldsymbol{y}_o,\boldsymbol{y}_c|\boldsymbol{\lambda})d\boldsymbol{y}_c}, \tag{1.14}$$

where the denominator equals to $p(\boldsymbol{y}_o|\boldsymbol{\lambda})\int_C p(\boldsymbol{y}_c|\boldsymbol{y}_o,\boldsymbol{\lambda})d\boldsymbol{y}_c$, and the numerator can be computed as

$$\begin{aligned}
&\int_{\mathbb{R}}\int_C y_{new}p(y_{new},\boldsymbol{y}_o,\boldsymbol{y}_c|\boldsymbol{\lambda})dy_{new}d\boldsymbol{y}_c\\
=&p(\boldsymbol{y}_o|\boldsymbol{\lambda})\int_C\{\int_{\mathbb{R}} y_{new}p(y_{new}|\boldsymbol{y}_o,\boldsymbol{y}_c,\boldsymbol{\lambda})dy_{new}\}p(\boldsymbol{y}_c|\boldsymbol{y}_o,\boldsymbol{\lambda})d\boldsymbol{y}_c\\
=&p(\boldsymbol{y}_o|\boldsymbol{\lambda})\int_C\{f(\boldsymbol{x}_{new},\boldsymbol{\lambda})+\hat{\tau}^2\boldsymbol{r}^T(\boldsymbol{x}_{new})\boldsymbol{\Sigma}^{-1}(\boldsymbol{y}-\boldsymbol{f}(\boldsymbol{\lambda})\}p(\boldsymbol{y}_c|\boldsymbol{y}_o,\boldsymbol{\lambda})d\boldsymbol{y}_c\\
=&p(\boldsymbol{y}_o|\boldsymbol{\lambda})\int_C(\boldsymbol{t}_{new}^T\boldsymbol{y}_c+s_{new}(\boldsymbol{\lambda}))p(\boldsymbol{y}_c|\boldsymbol{y}_o,\boldsymbol{\lambda})d\boldsymbol{y}_c\\
=&\boldsymbol{t}_{new}^T p(\boldsymbol{y}_o|\boldsymbol{\lambda})\int_C \boldsymbol{y}_c p(\boldsymbol{y}_c|\boldsymbol{y}_o,\boldsymbol{\lambda})d\boldsymbol{y}_c + s_{new}(\boldsymbol{\lambda})p(\boldsymbol{y}_o|\boldsymbol{\lambda})\int_C p(\boldsymbol{y}_c|\boldsymbol{y}_o,\boldsymbol{\lambda})d\boldsymbol{y}_c,
\end{aligned}$$

where

$$\begin{aligned}
&\boldsymbol{t}_{new}^T = \hat{\tau}^2\boldsymbol{r}^T(\boldsymbol{x}_{new})\boldsymbol{\Sigma}^{-1}_{[,n_o+1:n]},\\
&s_{new}(\boldsymbol{\lambda}) = f(\boldsymbol{x}_{new},\boldsymbol{\lambda}) - \boldsymbol{t}_{new}^T\boldsymbol{f}_c(\boldsymbol{\lambda}) + \hat{\tau}^2\boldsymbol{r}^T(\boldsymbol{x}_{new})\boldsymbol{\Sigma}^{-1}_{[,1:n_o]}(\boldsymbol{y}_o-\boldsymbol{f}_o(\boldsymbol{\lambda})).
\end{aligned}$$

By substituting the numerator and denominator into (1.14), we obtain

$$E(y_{new}|\mathcal{D},\boldsymbol{\lambda}) = \boldsymbol{t}_{new}^T E(\boldsymbol{y}_c|\mathcal{D},\boldsymbol{\lambda}) + s_{new}(\boldsymbol{\lambda}). \tag{1.15}$$

Taking expectation with respect to posterior distribution of $\boldsymbol{\lambda}$ for the formula above yields

$$E(y_{new}|\mathcal{D}) = \boldsymbol{t}_{new}^T E_{\boldsymbol{\lambda}|\mathcal{D}}\{E(\boldsymbol{y}_c|\mathcal{D},\boldsymbol{\lambda})\} + E_{\boldsymbol{\lambda}|\mathcal{D}} s_{new}(\boldsymbol{\lambda}),$$

which can be rewritten as (1.11).

A2. Expectation of multivariate truncated normal distribution

Suppose $\boldsymbol{W} \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, and $diag(\boldsymbol{\Sigma}) = (\sigma_1^2, \sigma_2^2, ..., \sigma_m^2)^T$. Then the joint distribution of $X_i \triangleq (W_i - \mu_i)/\sigma_i$ is $\boldsymbol{X} \sim N(\boldsymbol{0}, \boldsymbol{R})$, where $\boldsymbol{R}$ is the correlation matrix of $\boldsymbol{W}$, $i = 1, 2, ..., m$. Let $\phi_m(\boldsymbol{x}; \boldsymbol{R})$ be the joint density function of $\boldsymbol{X}$ which is given by

$$\phi_m(\boldsymbol{x}; \boldsymbol{R}) = \frac{1}{(2\pi)^{m/2}|\boldsymbol{R}|^{1/2}} \exp\{-\frac{1}{2}\boldsymbol{x}'\boldsymbol{R}^{-1}\boldsymbol{x}\}.$$

Let $C = [a_1, b_1]\times[a_2, b_2]\times...\times[a_m, b_m]$, and define $\Phi_m(\boldsymbol{a}, \boldsymbol{b}; \boldsymbol{R}) \triangleq \int_{a_1}^{b_1} ... \int_{a_m}^{b_m} \phi_m(\boldsymbol{x}; \boldsymbol{R}) dx_1 ... dx_m$. Then it can be shown that

$$E(X_i|\boldsymbol{X} \in C) = \alpha^{-1} \sum_{q=1}^{m} \rho_{iq}\{\phi(a_q)\Phi_{m-1}(\boldsymbol{A}_{q,a}, \boldsymbol{B}_{q,a}; \boldsymbol{R}_q) - \phi(b_q)\Phi_{m-1}(\boldsymbol{A}_{q,b}, \boldsymbol{B}_{q,b}; \boldsymbol{R}_q)\}, \tag{1.16}$$

where $\alpha = P(\boldsymbol{X} \in C) = \Phi_m(\boldsymbol{a}, \boldsymbol{b}; \boldsymbol{R})$, ρ_{iq} is the (i, q) element of correlation matrix $\boldsymbol{R}$, $\phi(a_q)$ is the standard normal density function evaluated at a_q, $\boldsymbol{A}_{q,y}$ is an $(m-1)\times 1$ vector with sth entry $A_{qs,y} = (a_s - \rho_{sq}y_q)/\sqrt{1-\rho_{sq}^2}$, $\boldsymbol{B}_{q,y}$ is an $(m-1)\times 1$ vector with sth entry $B_{qs,y} = (b_s - \rho_{sq}y_q)/\sqrt{1-\rho_{sq}^2}$, $y = a, b$, and $\boldsymbol{R}_q$ is the matrix of first-order partial correlation coefficients of X_s for $s \neq q$. Based on this we can readily obtain the expectation of truncated version of $\boldsymbol{W}$ by transforming $\boldsymbol{X}$ back to $\boldsymbol{W}$.

A3. Variance of the predictive distribution

The second moment conditional on the data and $\boldsymbol{\lambda}$ is given by

$$E(y_{new}^2|\mathcal{D},\boldsymbol{\lambda}) = \frac{\int_{\mathbb{R}}\int_C y_{new}^2 p(y_{new},\boldsymbol{y}_o,\boldsymbol{y}_c|\boldsymbol{\lambda})dy_{new}d\boldsymbol{y}_c}{\int_C p(\boldsymbol{y}_o,\boldsymbol{y}_c|\boldsymbol{\lambda})d\boldsymbol{y}_c},$$

where the denominator equals to $p(\boldsymbol{y}_o|\boldsymbol{\lambda})\int_C p(\boldsymbol{y}_c|\boldsymbol{y}_o,\boldsymbol{\lambda})d\boldsymbol{y}_c$, and the numerator can be computed as

$$\begin{aligned}
&\int_{\mathbb{R}}\int_C y_{new}p(y_{new},\boldsymbol{y}_o,\boldsymbol{y}_c|\boldsymbol{\lambda})dy_{new}d\boldsymbol{y}_c\\
=&p(\boldsymbol{y}_o|\boldsymbol{\lambda})\int_C\{\int_{\mathbb{R}} y_{new}^2 p(y_{new}|\boldsymbol{y}_o,\boldsymbol{y}_c,\boldsymbol{\lambda})dy_{new}\}p(\boldsymbol{y}_c|\boldsymbol{y}_o,\boldsymbol{\lambda})d\boldsymbol{y}_c\\
=&p(\boldsymbol{y}_o|\boldsymbol{\lambda})\int_C\{[E(y_{new}|\boldsymbol{y}_o,\boldsymbol{y}_c,\boldsymbol{\lambda})]^2+var(y_{new}|\boldsymbol{y}_o,\boldsymbol{y}_c,\boldsymbol{\lambda})\}p(\boldsymbol{y}_c|\boldsymbol{y}_o,\boldsymbol{\lambda})d\boldsymbol{y}_c\\
=&p(\boldsymbol{y}_o|\boldsymbol{\lambda})\int_C\{(\boldsymbol{t}_{new}^T\boldsymbol{y}_c+s_{new}(\boldsymbol{\lambda}))^2+v_{new}\}p(\boldsymbol{y}_c|\boldsymbol{y}_o,\boldsymbol{\lambda})d\boldsymbol{y}_c\\
=&\Big\{\boldsymbol{t}_{new}^T\left[\tilde{\boldsymbol{\Sigma}}_c(\boldsymbol{\lambda})+\tilde{\boldsymbol{f}}_c(\boldsymbol{\lambda})\tilde{\boldsymbol{f}}_c^T(\boldsymbol{\lambda})\right]\boldsymbol{t}_{new}+2s_{new}(\boldsymbol{\lambda})\boldsymbol{t}_{new}^T\tilde{\boldsymbol{f}}_c(\boldsymbol{\lambda})\\
&+s_{new}(\boldsymbol{\lambda})^2+v_{new}\Big\}p(\boldsymbol{y}_o|\boldsymbol{\lambda})\int_C p(\boldsymbol{y}_c|\boldsymbol{y}_o,\boldsymbol{\lambda})d\boldsymbol{y}_c.
\end{aligned}$$

Here

$$v_{new} = \hat{\tau}^2+\hat{\sigma}^2-\hat{\tau}^4\boldsymbol{r}^T(\boldsymbol{x}_{new})\boldsymbol{\Sigma}^{-1}\boldsymbol{r}(\boldsymbol{x}_{new}).$$

Thus the variance of the predictive distribution can be computed using

$$\begin{aligned}
&var(y_{new}|\mathcal{D},\boldsymbol{\lambda})\\
=&E(y_{new}^2|\mathcal{D},\boldsymbol{\lambda})-E^2(y_{new}|\mathcal{D},\boldsymbol{\lambda})\\
=&\boldsymbol{t}_{new}^T\left[\tilde{\boldsymbol{\Sigma}}_c(\boldsymbol{\lambda})+\tilde{\boldsymbol{f}}_c(\boldsymbol{\lambda})\tilde{\boldsymbol{f}}_c^T(\boldsymbol{\lambda})\right]\boldsymbol{t}_{new}+2s_{new}(\boldsymbol{\lambda})\boldsymbol{t}_{new}^T\tilde{\boldsymbol{f}}_c(\boldsymbol{\lambda})\\
&+s_{new}^2(\boldsymbol{\lambda})+v_{new}-\left[\boldsymbol{t}_{new}^T\tilde{\boldsymbol{f}}_c(\boldsymbol{\lambda})+s_{new}(\boldsymbol{\lambda})\right]^2\\
=&\boldsymbol{t}_{new}^T\tilde{\boldsymbol{\Sigma}}_c(\boldsymbol{\lambda})\boldsymbol{t}_{new}+v_{new}. \qquad (1.17)
\end{aligned}$$

Then, we incorporate the variability of $\boldsymbol{\lambda}$ using the law of total variance

$$var(y_{new}|\mathcal{D}) = E_{\boldsymbol{\lambda}|\mathcal{D}} var(y_{new}|\mathcal{D}, \boldsymbol{\lambda}) + var_{\boldsymbol{\lambda}|\mathcal{D}} E(y_{new}|\mathcal{D}, \boldsymbol{\lambda}).$$

Using the results from (1.15) and (1.17) we can easily verifies (1.12).

A4. Variance of multivariate truncated normal distribution

Using the same notations from Appendix A2, the formula for the second order moments can be obtained as

$$\begin{aligned}
&E(X_i X_j | \boldsymbol{X} \in C) \\
=&\rho_{ij} + \alpha^{-1} \sum_{q=1}^{m} \rho_{iq}\rho_{jq}\{a_q \phi(a_q)\Phi_{m-1}(\boldsymbol{A}_{q,a}, \boldsymbol{B}_{q,a}; \boldsymbol{R}_q) - b_q \phi(b_q)\Phi_{m-1}(\boldsymbol{A}_{q,b}, \boldsymbol{B}_{q,b}; \boldsymbol{R}_q)\} + \\
&\alpha^{-1} \sum_{q=1}^{m} \{\rho_{iq} \sum_{r \neq q} (\rho_{jr} - \rho_{jq}\rho_{qr})[\phi(a_q, a_r; \rho_{qr})\Phi_{m-2}(\boldsymbol{A}_{qr,aA}, \boldsymbol{B}_{qr,aA}; \boldsymbol{R}_{qr}) + \\
&\phi(b_q, b_r; \rho_{qr})\Phi_{m-2}(\boldsymbol{A}_{qr,bB}, \boldsymbol{B}_{qr,bB}; \boldsymbol{R}_{qr}) - \phi(a_q)\phi(B_{qr,a})\Phi_{m-2}(\boldsymbol{A}_{qr,aB}, \boldsymbol{B}_{qr,aB}; \boldsymbol{R}_{qr}) - \\
&\phi(b_q)\phi(A_{qr,b})\Phi_{m-2}(\boldsymbol{A}_{qr,bA}, \boldsymbol{B}_{qr,bA}; \boldsymbol{R}_{qr})]\}, i, j = 1, ..., m, \qquad (1.18)
\end{aligned}$$

where $\phi(a_q, a_r; \rho_{qr})$ is the bivariate normal density function with mean zero and covariance matrix $[1, \rho_{qr}; \rho_{qr}, 1]$ evaluated at (a_q, a_r), $\boldsymbol{A}_{qr,yY}$ is an $(m-2) \times 1$ vector with sth element $A^{(s)}_{qr,yY} = (A_{qs,y} - \rho_{sr.q} Y_{qr,y})/\sqrt{(1 - \rho^2_{sr.q})}$, $\rho_{sr.q} = (\rho_{rs} - \rho_{qs}\rho_{qr})/\sqrt{(1 - \rho^2_{qr})(1 - \rho^2_{qs})}$ with y taking a or b, Y taking A or B, and $\boldsymbol{R}_{qr}$ is the matrix of second order partial correlation coefficients of X_s for $s \neq q, s \neq r, q \neq r$. $\boldsymbol{B}_{qr,yY}$ is defined in the same way as $\boldsymbol{A}_{qr,yY}$. Thus the variance-covariance matrix of the truncated normal distribution can be computed as

$$Var(\boldsymbol{X}|\boldsymbol{X} \in C) = E(\boldsymbol{X}\boldsymbol{X}^T|\boldsymbol{X} \in C) - E(\boldsymbol{X}|\boldsymbol{X} \in C)E(\boldsymbol{X}|\boldsymbol{X} \in C)^T.$$

A5. Approximate method based on a mixture of normal distributions

When approximating the censored region via the normal distribution is inadequate, we may adopt the mixture normal distribution to achieve a better approximation. Again, consider the case of a uniform distribution $U(a, b)$. Let $m(y; \boldsymbol{\mu}, \boldsymbol{\nu}, \boldsymbol{w})$ denote the density of the mixture normal distribution with K components such that

$$m(y; \boldsymbol{\mu}, \boldsymbol{\nu}, \boldsymbol{w}) = \sum_{k=1}^{K} w_k \phi(y; \mu_k, \nu_k),$$

where $\sum_{k=1}^{K} w_k = 1$. We obtain the optimal parameter set by minimizing the Hellinger distance, i.e.,

$$\begin{aligned}(\boldsymbol{\mu}^*, \boldsymbol{\nu}^*, \boldsymbol{w}^*) &= \arg\min_{\boldsymbol{\mu},\boldsymbol{\nu},\boldsymbol{w}} \int_{\mathbb{R}} \left\{ m^{1/2}(y; \boldsymbol{\mu}, \boldsymbol{\nu}, \boldsymbol{w}) - \frac{I_{(a,b)}(y)}{\sqrt{b-a}} \right\}^2 dy \\ &= \arg\max_{\boldsymbol{\mu},\boldsymbol{\nu},\boldsymbol{w}} \int_a^b m^{1/2}(y; \boldsymbol{\mu}, \boldsymbol{\nu}, \boldsymbol{w}) dy.\end{aligned}$$

Using normal mixture does not yield an explicit expression as before, so we need to optimize the parameters numerically. Exploitation of the symmetry of $m(y)$ helps reducing the number of parameters. Let $\mu_0 = (a + b)/2$. When the number of components $K = 2$, for instance, $\boldsymbol{\mu} = \mu_0 \pm \Delta, \nu_1 = \nu_2 = \nu$ and $w_1 = w_2 = 0.5$ so the number of parameters to estimate is reduced to two, that is, Δ and ν. Similarly we only need to estimate four parameters when $K = 3$. The two cases with $K = 2$ and $K = 3$ are shown in Figure 1.6.

Now that we have the optimal parameter set $(\boldsymbol{\mu}^*, \boldsymbol{\nu}^*, \boldsymbol{w}^*)$, we can use the same model as in (1.13) except that now μ_0 is treated as the response and e follows the mixture normal distribution with density $m(e; \boldsymbol{\mu}^* - \boldsymbol{\mu}_0, \boldsymbol{\nu}^*, \boldsymbol{w}^*)$ where $\boldsymbol{\mu}_0$ denote the vector of μ_0 with length K. We can still estimate the parameters of the model under Bayesian framework. Following the notations from Section 3.1 with $\boldsymbol{y}_c$ replaced by $\boldsymbol{\mu}_0$, the likelihood function

can be written as

$$L(\boldsymbol{\lambda}, \boldsymbol{\psi}; \boldsymbol{y}) = p(\boldsymbol{y}|\boldsymbol{\lambda}, \boldsymbol{\psi}) = \sum_{k=1}^{K} w_k \phi(\boldsymbol{y}; \boldsymbol{f}(\boldsymbol{\lambda}) + \boldsymbol{\mu}_k, \boldsymbol{\Sigma} + \boldsymbol{\Sigma}_k),$$

where $\boldsymbol{\mu}_k = ((0, ..., 0)_{1\times n_o}, (\mu_k^* - \mu_0, ..., \mu_k^* - \mu_0)_{1\times n_c})^T$ and $\boldsymbol{\Sigma}_k$ is the diagonal matrix with diagonal elements $((0, ..., 0)_{1\times n_o}, (\nu_k^*, ..., \nu_k^*)_{1\times n_c})$. We can conduct Bayesian inference of the parameters following the same strategy as Section 3.1.

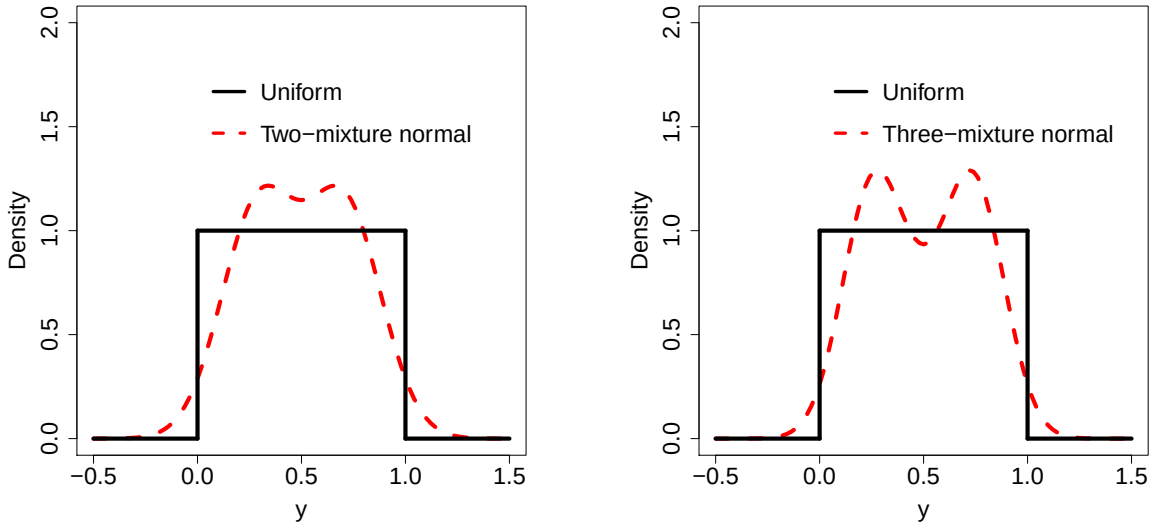

Figure 1.6: Two-mixture (left) and three-mixture(right) normal approximations of $U(0, 1)$.

The predictive distribution based on the normal mixture model is

$$p(y_{new}|\boldsymbol{y}) = \int p(y_{new}|\boldsymbol{y}, \boldsymbol{\lambda}) p(\boldsymbol{\lambda}|\boldsymbol{y}) d\boldsymbol{\lambda},$$

where

$$p(\boldsymbol{\lambda}|\boldsymbol{y}) = \frac{\pi(\boldsymbol{\lambda}) L(\boldsymbol{\lambda}, \hat{\boldsymbol{\psi}}; \boldsymbol{y})}{\int \pi(\boldsymbol{\lambda}) L(\boldsymbol{\lambda}, \hat{\boldsymbol{\psi}}; \boldsymbol{y}) d\boldsymbol{\lambda}},$$

$$p(y_{new}|\boldsymbol{y}, \boldsymbol{\lambda}) = \frac{p(y_{new}, \boldsymbol{y}|\boldsymbol{\lambda})}{p(\boldsymbol{y}|\boldsymbol{\lambda})},$$

and

$$p(y_{new}, \boldsymbol{y}|\boldsymbol{\lambda}) = \sum_{k=1}^{K} w_k \phi\Bigg(\begin{bmatrix} y_{new} \\ \boldsymbol{y} \end{bmatrix}; \begin{bmatrix} f(\boldsymbol{x}_{new}, \boldsymbol{\lambda}) \\ \boldsymbol{f}(\boldsymbol{\lambda}) + \boldsymbol{\mu}_k \end{bmatrix}, \begin{bmatrix} \hat{\tau}^2 + \hat{\sigma}^2 & \hat{\tau}^2 \boldsymbol{r}^T(\boldsymbol{x}_{new}) \\ \hat{\tau}^2 \boldsymbol{r}(\boldsymbol{x}_{new}) & \boldsymbol{\Sigma} + \boldsymbol{\Sigma}_k \end{bmatrix} \Bigg).$$

The predictive distribution above is not hard to evaluate and thus we can obtain predictive samples of y_{new} using Monte Carlo methods.

CHAPTER 2

AUTOMATIC KRIGING FOR LARGE DATASET

In this Chapter, we propose an interpolation technique which can be used with large and unstructured data. Kriging is widely used for interpolation of unstructured data because of its ability to produce confidence intervals for predictions. The model is fitted to the data using maximum likelihood or cross validation-based methods. Unfortunately, the fitting is expensive for large data because one evaluation of the objective function requires $O(n^3)$ operations, where n is the size of the data. There exist other interpolation techniques such as inverse distance weighting (IDW), which do not require any estimation and therefore can be easily used with large data. However, the performance of IDW can be significantly worse than kriging. In this chapter, we propose a kriging method that does not require any estimation from data and whose performance is much better than that of IDW. We also propose a novel approach to choose nuggets in kriging that can produce numerically stable results, which is important for applying the technique to unstructured data. A technique for adaptively choosing the kernels is also developed.

2.1 Introduction

Gaussian process or kriging is a commonly used technique to build surrogate models. It was originally applied in geostatistical problems [30]. Nowadays it has a wide range of applications in metamodeling, interpolation, spatial statistics, computer experiments, hydrogeology, remote sensing, etc. Among all the metamodeling techniques, kriging performs reasonably well in terms of accuracy and model complexity (see, e.g., [31]).

In the "big data" era, the scalability of statistical methods becomes more and more important. There is no exception for the Gaussian process predictors. For Gaussian process model, the correlation parameters are usually unknown and need to be estimated. The

classic estimation methods are either based on maximizing the likelihood or minimizing the cross-validation error, but they are all quite computationally intensive when the sample size n or the dimensionality p goes large. This is because both maximum-likelihood and cross-validation involve solving the inverse of correlation matrix of the design with a number of varying correlation parameters, which can be prohibitive for large dataset.

There is a rich literature on mitigating the computational complexity of estimation in Gaussian process model. [23] uses the predictive process model which projects the original process onto lower dimensional space to simplify the computations. On the other hand, [32] resorts to approximation of the likelihood. [33], [34], [35], [36], [37], [38] approximate the covariance function and make use of sparse covariance matrix. Neighborhood-based local Gaussian process methods have been proposed by [39], [40], [41], [42], and they focus mainly on making predictions. [43] and [9] adopt tree-based regression to deal with massive data. [44] models large-scale computer experiment through iterative construction of the predictor. [45] incorporates physical knowledge into the statistical model and proposes a surrogate model capable of conducting efficient predictions. Advanced computing techniques using GPU, cluster, symmetric-multiprocessor, etc. are also deployed in Gaussian process modeling, see [46] among others.

To circumvent the difficulty of solving matrix inverse, we propose the automatic kriging method which quickly finds the correlation parameter according to the mutual distances of design points. The computation using proposed method is quite simple and there is no need to evaluate the objective function involving matrix inversion for many times, as is the case in traditional methods. Since the distance-based estimation may not be as accurate as the likelihood/cross validation-based estimation, the predictor in proposed approach should be robust to misspecification of correlation parameters. Predictors with such properties can be found in [47], [48] among others. With a properly chosen correlation function, we can show that the automatic kriging predictor will converge to the inverse distance weighting predictor ([49], [50]) in the worst case.

Unstructured data often takes place in real-world applications. Namely, the observations do not necessarily come from well-designed experiments and there can be unnecessary replicates and close design points. The proposed predictor can experience numerical stability issues when the design points fall too close to each other. There exist several techniques to deal with such problems, such as adding nugget to the correlation matrix, covariance tapering [36], and fixed rank kriging [35], to list a few. Among them, adding nugget is the simplest approach yet yields satisfactory performance in most of the cases. Adding nugget can improve numerical stability and coverage of confidence interval, see [51], [52]. A drawback of using such method is that the predictor is no longer an interpolator, but a smoother instead. [53] derives the lower bound of constant nugget to minimize the unnecessary over-smoothing. [52] also studies how to choose the fixed nugget for kriging modeling.

The nugget term can be regarded as the error variance. As heteroskedasticity occurs often in statistical modeling, it is a natural idea to use a non-constant nugget. Such type of idea can be found in many works. [54] uses another Gaussian process to model the error variance of the Gaussian process regression. [55] proposes kriging with modified nugget effect which assumes heterogeneous error variance. The stochastic kriging proposed by [56] assumes that replicates are available at each design point and the error variance at a certain location is estimated from the sample variance of the replicates. [57] demonstrates through numerical examples the robustness of ordinary kriging in metamodeling with heterogeneous variances. [58] studies Gaussian process regression with input measurement error. [59] adopts heteroskedastic Gaussian process model to handle simulation experiment with replicates. We propose the adaptive nugget approach to solve the instability problem in unstructured data. The idea behind our method is to add nugget adaptively so as to control the determinant of covariance matrix and to improve stability of the predictor. For design points which are close to each other, the proposed method will automatically assign large nugget to them.

Automatic kriging can be further enhanced by having adaptive kernel. Constant correlation parameter across all data points is the standard assumption in usual Gaussian process models, but in real-world problems, the underlying process can be non-stationary and the correlation parameters can vary. The adaptive kernel approach proposed in this chapter is to provide Gaussian process model with the flexibility of dealing with complex and non-stationary problems. Non-stationary covariance functions has been studied by others as well. [60] proposes to model the covariance structure and build the kriging model using only the design points within a moving window centered at the estimate location. The Ph.D. thesis [61] proposes a covariance function with spatially varying length scales, but it is hard to carry out the estimation based on such covariance function. Non-stationary Gaussian process is also studied in [62], which proposes to map the original spatial processes into a latent space to deal with non-stationarity and anisotropy. [63] proposes a non-stationary kriging model with lifted Brownian coviance function which demonstrates desirable behavior in both short and long ranges. The adaptive kernel approach proposed in the chapter adopts distance-based kernel and avoids the need for estimation, which results in rapid computation.

The Chapter is organized as follows: Section 2.2 introduces the framework of Gaussian process modeling and the distance-based estimation for correlation parameter. It also describes he limiting behavior of automatic kriging. The proposed predictor is compared with inverse distance weighting and ordinary kriging through simulation studies. In Section 2.3, the performance of the automatic kriging is further enhanced by adding adaptive nugget. We illustrate the computation of individual nugget for each design point and iterative construction of correlation matrix inverse. The adaptive kernel idea is demonstrated in Section 2.4, where the estimation and the construction of the predictor are shown in detail. We draw the conclusions in Section 2.5.

2.2 Automatic Kriging

Suppose a certain system output y($\boldsymbol{x}$) can be represented using the Gaussian process

$$y(\boldsymbol{x}) = \mu + \delta(\boldsymbol{x}), \tag{2.1}$$

where μ is the unknown mean parameter, $\delta(\boldsymbol{x})$ follows a Gaussian process $GP(0, \tau^2 R(\cdot, \cdot))$, and τ^2 is the variance of the Gaussian process. Here $R(\cdot, \cdot)$ is the inverse multiquadric correlation function (see [64]) given by

$$R(\boldsymbol{x}, \boldsymbol{w}) = \frac{1}{1 + \sum_{j=1}^{p} (\frac{x_j - w_j}{\theta_j})^2}, \tag{2.2}$$

with $\boldsymbol{\theta} = (\theta_1, ..., \theta_p)^T$ being the correlation parameters. A systematic review of the correlation functions for Gaussian process can be found in [65]. Suppose we have n design points $\boldsymbol{X} = \{\boldsymbol{x}_i \in [0,1]^p, i = 1, ..., n\}$ and the corresponding responses are $\boldsymbol{y} = (y_1, ..., y_n)^T$. The traditional methods to estimate $\boldsymbol{\theta}$ are based on likelihood or cross-validation. These methods involve the inverse of $n \times n$ correlation matrix $\boldsymbol{R}$ whose ijth element is $R(\boldsymbol{x}_i, \boldsymbol{x}_j)$,$i, j = 1, ..., n$. The computation is very costly when n or p is large. Therefore, we aim to seek for approach with fast estimation which can still yield good fitting.

Our idea is to build such an estimator based on mutual distances among all design points. We first simplify the correlation function to be isotropic such that $\theta_1 = ... = \theta_p = \theta$. Then the correlation function can be written as $R(\boldsymbol{x}_i, \boldsymbol{x}_j) = 1/\{1 + d_{ij}^2/\theta^2\}$, where d_{ij} is the Euclidean distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, $i, j \in \{1, ..., n\}$. If we assume a certain type of "averaged" distance can be obtained, θ can be estimated by setting the correlation corresponding to the "averaged" distance to $1/2$. For each design point $\boldsymbol{x}_i$, define its filling

distance as

$$D_i = \max_{\boldsymbol{x}\in[0,1]^p} \{d(\boldsymbol{x}, \boldsymbol{x}_i) | d(\boldsymbol{x}, \boldsymbol{x}_i) < d(\boldsymbol{x}, \boldsymbol{x}_j), \forall j = 1, ..., n\},$$

where $d(\boldsymbol{x}, \boldsymbol{x}_i)$ is the Euclidean distance between $\boldsymbol{x}$ and $\boldsymbol{x}_i$. Then our estimate of $\boldsymbol{\theta}$ can be obtained through three approaches

$$\begin{aligned}
\hat{\theta} =& \text{median}_{i=1,...,n}\{D_i\}, \text{ or} \\
\hat{\theta} =& \text{max}_{i=1,...,n}\{D_i\}, \text{ or} \\
\hat{\theta} =& \text{quantile}\{D_i, \max(1 - \frac{10p}{2n}, 0.5), i = 1, ..., n\},
\end{aligned} \tag{2.3}$$

among which the quantile approach is of our final choice. The reasoning and empirical studies for using quantile to estimate θ are given in the Appendix in Section 2.6. The distance-based estimator for θ is much faster than those obtained from traditional methods, so it is an ideal candidate to perform Gaussian process modeling in a big data setting.

Although our correlation parameter estimate is easy to compute, it can be less accurate than the likelihood or cross validation-based estimates. Thus one important aspect of our methodology is to have an emulator which is robust to parameter misspecification. The limit kriging (LK) predictor ([47]) can serve as a good candidate. The LK predictor takes the form

$$\hat{y}_{LK}(\boldsymbol{x}) = \frac{\boldsymbol{r}^T(\boldsymbol{x})\boldsymbol{R}^{-1}\boldsymbol{y}}{\boldsymbol{r}^T(\boldsymbol{x})\boldsymbol{R}^{-1}\mathbf{1}}, \tag{2.4}$$

where $\boldsymbol{r}(\boldsymbol{x})$ is an n-dimensional vector with ith element $R(\boldsymbol{x}, \boldsymbol{x}_i)$, and $\mathbf{1}$ is the n-dimensional vector composed of ones. The widely-used inverse distance weighting (IDW) predictor is defined as

$$\hat{y}_{IDW}(\boldsymbol{x}) = \frac{\sum_{i=1}^n y_i/d^2(\boldsymbol{x}, \boldsymbol{x}_i)}{\sum_{i=1}^n 1/d^2(\boldsymbol{x}, \boldsymbol{x}_i)}.$$

We have the following connection between LK and IDW predictors:

Theorem 2.2.1. *Let the correlation function be the inverse multiquadric correlation defined in (2.2). Then as* $\theta \to 0$, $\hat{y}_{LK}(\boldsymbol{x}) \to \hat{y}_{IDW}(\boldsymbol{x})$.

Proof. Under inverse multiquadric kernel, the ith element of $\boldsymbol{r}(\boldsymbol{x})$ is $R(\boldsymbol{x}, \boldsymbol{x}_i) = 1/\{1 + d^2(\boldsymbol{x}, \boldsymbol{x}_i)/\theta^2\}$. As $\theta \to 0$, it can be shown that $\boldsymbol{R} \to \boldsymbol{I}$ where $\boldsymbol{I}$ is the identity matrix, and thus

$$\lim_{\theta \to 0} \hat{y}_{LK}(\boldsymbol{x}) = \lim_{\theta \to 0} \frac{\sum_{i=1}^{n} \frac{y_i}{\theta^2 + d^2(\boldsymbol{x}, \boldsymbol{x}_i)}}{\sum_{i=1}^{n} \frac{1}{\theta^2 + d^2(\boldsymbol{x}, \boldsymbol{x}_i)}} = \hat{y}_{IDW}(\boldsymbol{x}).$$

□

On the other hand, the ordinary kriging (OK) predictor does not share such good property. Similar to the proof of Theorem 2.2.1, we can show that

$$\lim_{\theta \to 0} \hat{y}_{OK}(\boldsymbol{x}) = \begin{cases} \bar{y} & , \boldsymbol{x} \notin \boldsymbol{X}, \\ y(\boldsymbol{x}) & , \boldsymbol{x} \in \boldsymbol{X}, \end{cases}$$

which makes OK predictor very volatile when θ goes to zero. If we look at Figure 2.1 where IDW, OK and LK are compared using function $y = \sin[30(x - 0.9)^4] \cos(2x - 1.8) + (x - 0.9)/2$ in [66] with 20 uniformly distributed design points, LK predictor performs better than both IDW and OK with distance-based correlation parameter estimate.

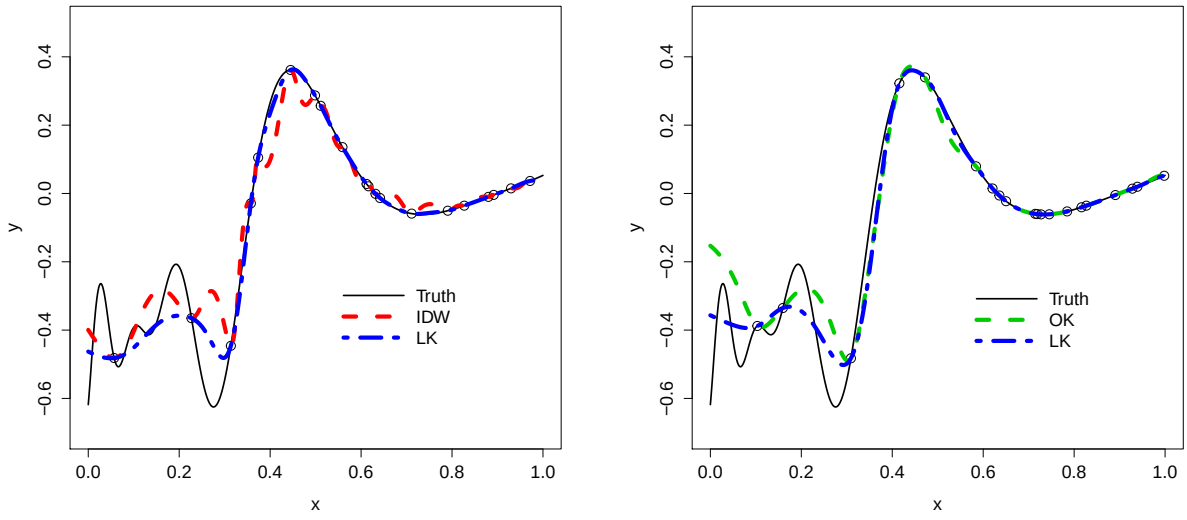

Figure 2.1: The comparison of IDW, OK and LK with $n = 20$

We conduct a formal simulation study to check the performance of different predictors. Within each simulation run, we generate designs with 20 and 100 points from $U(0, 1)$, respectively. Inverse distance weighting is used as benchmark and the MSPE of LK and OK are compared with that of IDW. The MSPE for each predictor is computed as

$$\text{MSPE} = \frac{1}{n_{test}} \sum_{i=1}^{n_{test}} (\hat{y}(\boldsymbol{x}_i) - y(\boldsymbol{x}_i))^2,$$

where n_{test} is the size of the testing data. There are 100 simulations runs with each design size. The ratios of MSPE are summarized and displayed in Figure 2.2. OK outperforms IDW in 20-point design, but not the case in 100-point design. On the contrary, automatic kriging using LK consistently performs well regardless of the design size. Thus the limit kriging predictor is of our choice for the proposed methodology.

Now we compare the performance of different predictors using higher dimensional examples. We simulate design points from uniform distribution on unit hypercube $[0, 1]^p$, $p >$

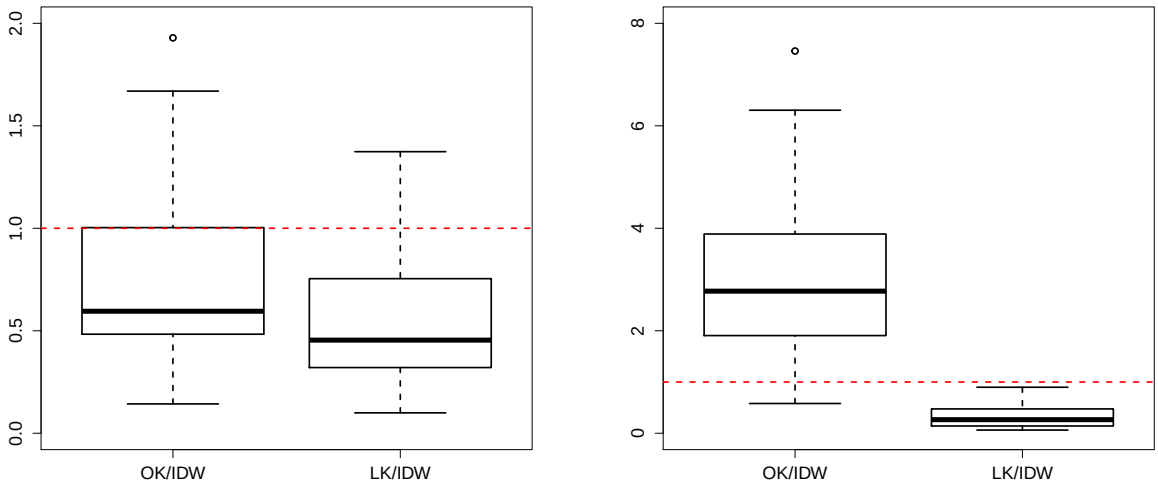

Figure 2.2: The comparison of IDW, OK, LK with $n = 20$ (left) and $n = 100$ (right) uniformly-distributed design points using 1d function.

1. The Franke two-dimensional function

$$\begin{aligned} y =& 0.75\exp\{-\frac{(9x_1-2)^2}{4} - \frac{(9x_2-2)^2}{4}\} + 0.75\exp\{-\frac{(9x_1+1)^2}{49} - \frac{9x_2+1}{10}\} \\ &+ 0.5\exp\{-\frac{(9x_1-7)^2}{4} - \frac{(9x_2-3)^2}{4}\} - 0.2\exp\{-(9x_1-4)^2 - (9x_2-7)^2\}, \end{aligned}$$

and the eight-dimensional Borehole function

$$y = \frac{2\pi T_u(H_u - H_l)}{\ln(r/r_w)\{1 + \frac{2LT_u}{\ln(r/r_w)r_w^2 K_w} + T_u/T_l\}}$$

are adopted to compare the methods. The fit by IDW and LK from one simulation run is displayed in Figure 2.3. According to the results with 100 simulations in Figure 2.4, automatic kriging with LK still achieves the best performance among all three methods. Thus the superiority of automatic kriging is demonstrated for uniformly-distributed designs.

Automatic kriging avoids the costly likelihood/cross validation-based estimation, so it should have advantage in handling large dataset. In fact, it not only bypasses inverting

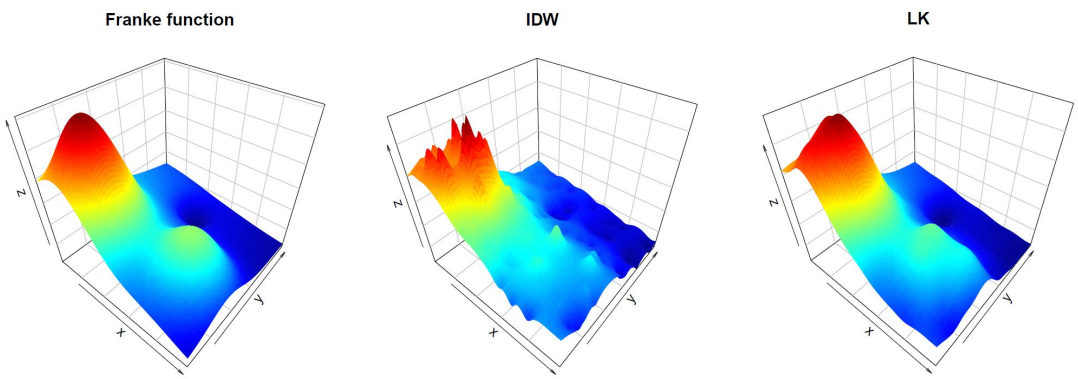

Figure 2.3: Comparing IDW and LK in 2d example. Left panel: Franke 2d function. Middle panel:The fit by IDW using 100 design points from $U(0,1)^2$. Right panel: The fit by LK using 100 design points from $U(0,1)^2$

the high dimensional correlation matrix, but also makes the computational complexity irrelevant to p since all dimensions share the common distance-based correlation parameter. For large designs, automatic kriging using LK still performs well in terms of overall fit as shown in Figure 2.5. Moreover, it is much faster than the approaches using traditional estimation. For example, one simulation run using Borehole function with 1,000 design and 1,000 testing points takes automatic kriging about 10 seconds while it takes 2400s for a 300-point design with 1000 testing points using likelihood-based method implemented via the *GPfit* library in R on a 3.33 GHz desktop.

We also compare the computational time of automatic kriging with other state-of-art methods. For this comparison, we employ the Ackley's path function

$$y = -a\exp\left\{-b\sqrt{\sum_{i=1}^{p} x_i^2/p}\right\} - \exp\{\sum_{i=1}^{p}\cos(cx_i)/p\} + a + \exp(1), \boldsymbol{x} \in [-2,2]^p$$

with $a = 2p, b = 0.2$ and $c = 2\pi$. The plot of the function when $p = 2$ is shown in Figure 2.6. The performance of automatic kriging is compared with that of local Gaussian process proposed in [39] implemented via R package *laGP*. When there are 1,000 design points and 1,000 prediction points with $p = 50$, the estimation and prediction altogether take

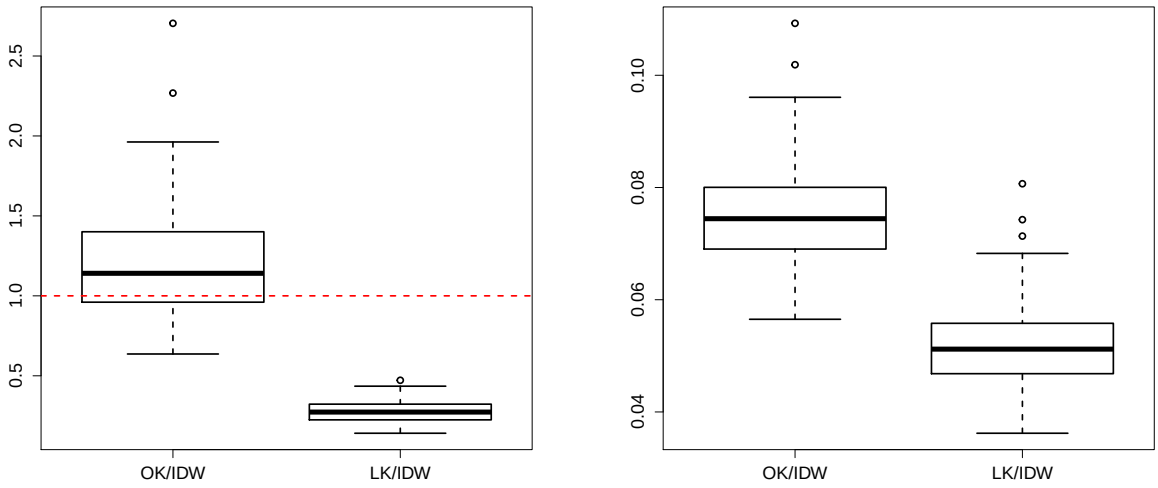

Figure 2.4: Comparing the MSPE of IDW, OK and LK. Left panel: Franke 2d function with 100 uniformly-distributed design points. Right panel: Borehole function with 300 uniformly-distributed design points.

19.3 seconds for automatic kriging and 21.4 seconds for local Gaussian process, and their MSPE are 0.33 and 14.4, respectively. Let m denote the size of the neighborhood in local Gaussian process. The theoretical complexity of the two methods are $O(n^3 + n_{test}n)$ and $O(n_{test}m^3)$, respectively. Then we systematically compare the two methods with different design size n and dimensionality p, and the results are summarized in Table 2.1. We did not vary the number of prediction points since the computational complexity of both methods are linear in the number of prediction points. Automatic kriging demonstrates better fit to the data than local Gaussian process, and the computational time of the two methods is comparable with moderate design size. When n is much larger than the neighborhood size in local Gaussian process, *laGP* is faster than automatic kriging.

2.3 Adaptive Nugget

Oftentimes there exist close points in real data, especially when the size of dataset is large. In such cases, the correlation matrix $\boldsymbol{R}$ is ill-conditioned and the performance of

Figure 2.5: Comparing the MSPE of IDW, OK and LK for large designs. Left panel: OK vs. IDW for 1d function with 10,000 design points. Middle panel:LK vs. IDW for 1d function with 10,000 design points. Right panel: Borehole 8d function with 10,000 design points.

Table 2.1: The comparison between automatic kriging and local Gaussian process using Ackley's path function

	Automatic kriging		local GP	
	Time(s)	MSPE	Time(s)	MSPE
$n = 100, p = 10$	1.75	0.21	1.28	1.02
$n = 1000, p = 10$	15.53	0.08	15.94	1.51
$n = 10000, p = 10$	340.96	0.08	16.75	1.43
$n = 100, p = 50$	2.27	1.65	1.81	8.06
$n = 1000, p = 50$	19.3	0.33	21.38	14.4
$n = 10000, p = 50$	417.45	0.03	23.08	25.44

both LK and OK deteriorate quickly (see the left panel of Figure 2.7, where there is only one pair of close points in red circle). It has been a common practice to add a fixed nugget $\lambda \boldsymbol{I}$ to $\boldsymbol{R}$ to mitigate such problem where $\lambda > 0$ and $\boldsymbol{I}$ is the $n \times n$ identity matrix (see [51] among others). The nuggets can be viewed as the error variance of the corresponding observations and they do not necessarily remain the same. For example, large nuggets are added to close points but very small or even no nuggets are needed for other points. As can be observed from right panel of Figure 2.7, there is only one pair of close points but adding fixed nugget will drag all the points towards the mean.

The method proposed by us can actually assign different nuggets to different locations.

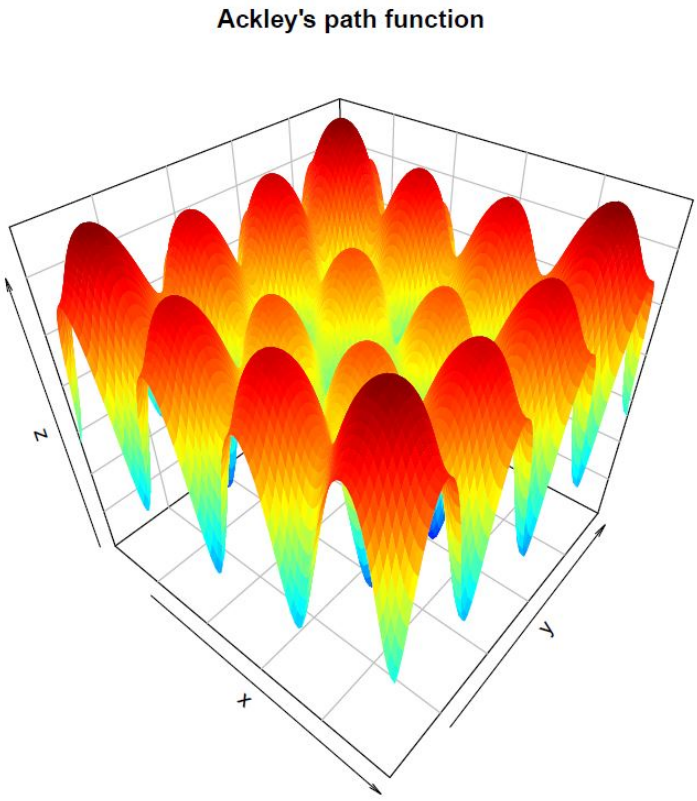

Figure 2.6: The plot of Ackley's path function when $p = 2$

Our principle is to sequentially add point to the current design and then compute its nugget. Suppose the design points are ordered (we shall show how to decide the order later) and we have already added nuggets for k points. Now, we want to choose the nugget for the next point $\boldsymbol{x}_{k+1}$. The principle for computing the nugget is to stabilize our predictor by controlling the determinant of the working correlation matrix $\boldsymbol{R} + \boldsymbol{\Lambda}$ where $\boldsymbol{\Lambda}$ is the diagonal matrix with diagonal elements $\lambda_1, ..., \lambda_n$ which we call a nugget matrix. Let $\boldsymbol{R}_k$ and $\boldsymbol{\Lambda}_k$ denote the correlation and nugget matrix for the first k points. Now when there are $k + 1$ points,

$$\boldsymbol{R}_{k+1} + \boldsymbol{\Lambda}_{k+1} = \begin{bmatrix} \boldsymbol{R}_k + \boldsymbol{\Lambda}_k & \boldsymbol{r}_k(\boldsymbol{x}_{k+1}) \\ \boldsymbol{r}_k^T(\boldsymbol{x}_{k+1}) & 1 + \lambda_{k+1} \end{bmatrix}.$$

We can show the existence of a sequence of nugget to control the determinant of working correlation matrix, as stated in the theorem below.

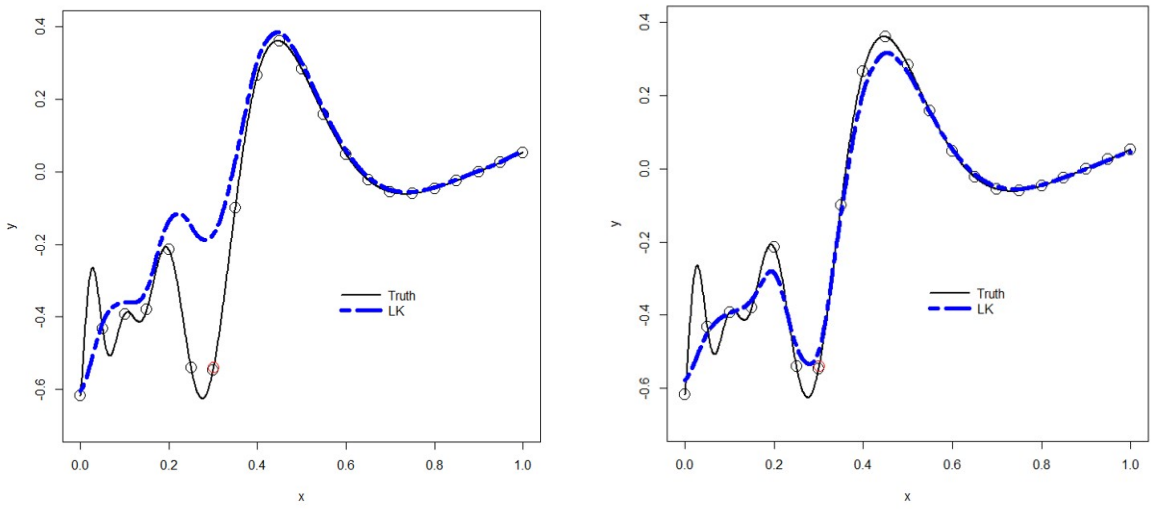

Figure 2.7: Design with close points. Left panel: LK fails when having one pair of close points. Right panel: Using constant nugget will pull all the design points towards the mean.

Theorem 2.3.1. *There exists a sequence of nugget* $\lambda_1, \lambda_2, ..., \lambda_n$ *such that* $|\boldsymbol{R}_k + \boldsymbol{\Lambda}_k|$ *is constant for* $k = 1, ..., n$.

Proof. Based on the result of determinant of block matrix, we have

$$|\boldsymbol{R}_{k+1}+\boldsymbol{\Lambda}_{k+1}| = \{1+\lambda_{k+1}-\boldsymbol{r}_k^T(\boldsymbol{x}_{k+1})(\boldsymbol{R}_k+\boldsymbol{\Lambda}_k)^{-1}\boldsymbol{r}_k(\boldsymbol{x}_{k+1})\}|\boldsymbol{R}_k+\boldsymbol{\Lambda}_k|, k = 1, ..., n-1,$$

where $\boldsymbol{r}_k(\boldsymbol{x})$ is a k-dimensional vector with ith element $R(\boldsymbol{x}_i, \boldsymbol{x}), i = 1, ..., k$. If we compute the nugget for $k+1$st point by

$$\lambda_{k+1} = \boldsymbol{r}_k^T(\boldsymbol{x}_{k+1})(\boldsymbol{R}_k + \boldsymbol{\Lambda}_k)^{-1}\boldsymbol{r}_k(\boldsymbol{x}_{k+1}), \tag{2.5}$$

the determinant will remain constant. By further setting $\boldsymbol{\Lambda}_1 = \lambda_1 = 0$, $|\boldsymbol{R}_1 + \boldsymbol{\Lambda}_1| = ... = |\boldsymbol{R}_n + \boldsymbol{\Lambda}_n| = |\boldsymbol{R} + \boldsymbol{\Lambda}| = 1$. □

After enabling adaptive nugget, as one may observe from Figure 2.8 and 2.9, the condition number of the correlation matrix remains small through controlling the determinant.

In this example, we use the one-dimensional function $y = \exp\{(x+1/2)^2\} \cdot \sin\{\exp[(x+1/2)^2]\}$ from [44] with 81 evenly spaced design points and the correlation parameter is fixed at 269.

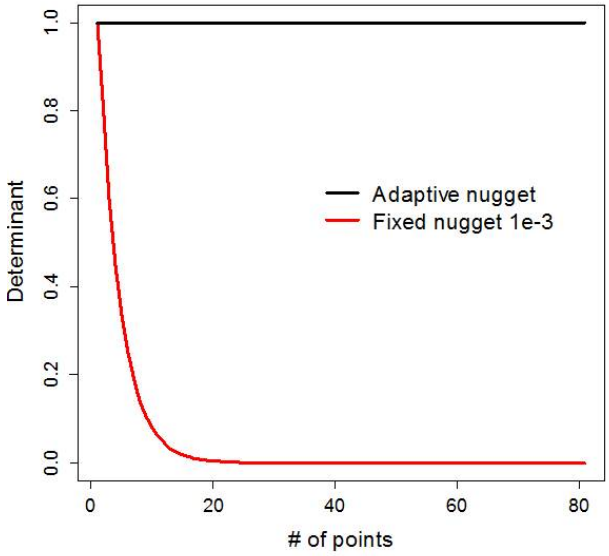

Figure 2.8: The determinant from fixed nugget approach versus that from adaptive nugget approach

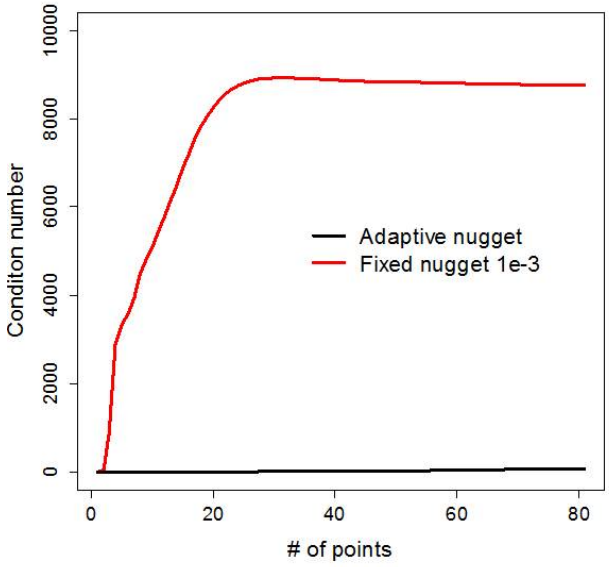

Figure 2.9: The condition number from fixed nugget approach versus that from adaptive nugget approach

If we simply use (2.5) and make the determinant remain constant, the nuggets tend to be unnecessarily large which leads to over-smoothed emulator. Hence one more algorithmic parameter q is added for computing the nugget of a newly added design point to avoid the nugget being too large

$$\lambda_{k+1} = \frac{1}{(1-q)^3}\{\boldsymbol{r}_k^T(\boldsymbol{x}_{k+1})(\boldsymbol{R}_k + \boldsymbol{\Lambda}_k)^{-1}\boldsymbol{r}_k(\boldsymbol{x}_{k+1}) - q\}_+^3, \tag{2.6}$$

where $\boldsymbol{r}_k(\boldsymbol{x}_{k+1})$ is a k-dimensional vector with ith element $R(\boldsymbol{x}_i, \boldsymbol{x}_{k+1}), i = 1, ..., k$, and the parameter q is to control the size of nugget. A rule of thumb is to use $q = 0.99$ so that a nugget is not added unless $\boldsymbol{r}_k^T(\boldsymbol{x}_{k+1})(\boldsymbol{R}_k + \boldsymbol{\Lambda}_k)^{-1}\boldsymbol{r}_k(\boldsymbol{x}_{k+1}) > q$. A comparison of computing nugget through (2.5) and (2.6) is made in the left panel of Figure 2.10.

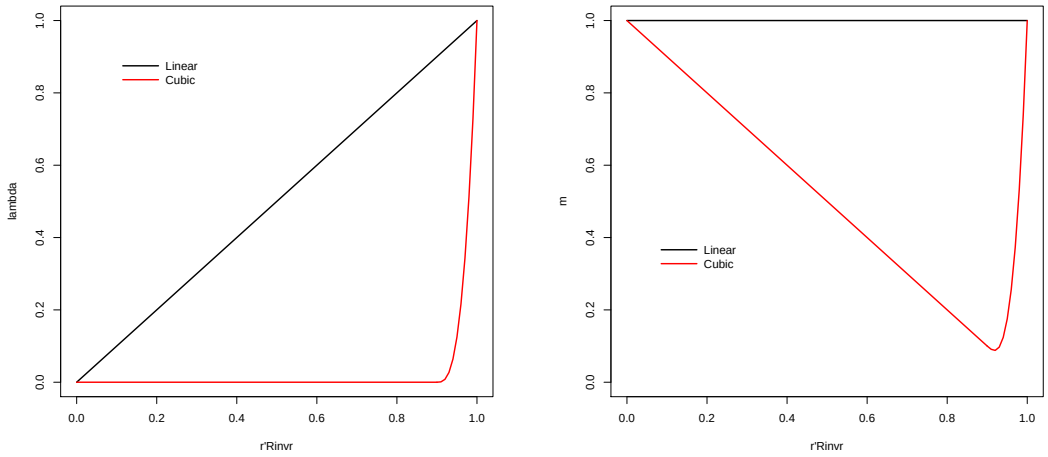

Figure 2.10: The comparison of computing nugget using formula (2.5) and (2.6). Left panel: Plot of λ for newly added point. Right panel: Plot of m in formula (2.8).

Now we show how to decide the order of the design. The first point $\boldsymbol{x}_{(1)}$ is chosen to be $\boldsymbol{x}_{(1)} = arg\max_{\boldsymbol{x}_i}\{|y(\boldsymbol{x}_i) - \bar{y}|, i = 1, ..., n\}$ where $\bar{y} = \frac{1}{n}\sum_{i=1}^n y_i$. Suppose we have already ordered k points $\boldsymbol{x}_{(1)}, ..., \boldsymbol{x}_{(k)}, k = 1, ..., n-1$ and computed their corresponding

nuggets according to (2.6). Then the next point $\boldsymbol{x}_{(k+1)}$ is decided upon

$$\boldsymbol{x}_{(k+1)} = arg\min_{\boldsymbol{x}_i}\{\boldsymbol{r}_k^T(\boldsymbol{x}_i)(\boldsymbol{R}_k + \boldsymbol{\Lambda}_k)^{-1}\boldsymbol{r}_k(\boldsymbol{x}_i), \boldsymbol{x}_i \notin \boldsymbol{X}_k, i = 1, ..., n\}, \tag{2.7}$$

where $\boldsymbol{X}_k = \{\boldsymbol{x}_{(1)}, ..., \boldsymbol{x}_{(k)}\}$. The idea behind this formula is to choose next point with the smallest nugget to avoid the instability issue. Combined with formula (2.6), this enables us to obtain the fully ordered data $\boldsymbol{x}_{(1)}, ..., \boldsymbol{x}_{(n)}$ and the nugget matrix $\boldsymbol{\Lambda}$ simultaneously.

The limit kriging predictor with adaptive nugget can be written as

$$\hat{y}_{LKAN}(\boldsymbol{x}) = \frac{\boldsymbol{r}^T(\boldsymbol{x})\big(\boldsymbol{R} + \boldsymbol{\Lambda}\big)^{-1}\boldsymbol{y}}{\boldsymbol{r}^T(\boldsymbol{x})\big(\boldsymbol{R} + \boldsymbol{\Lambda}\big)^{-1}\boldsymbol{1}}.$$

As mentioned in the first paragraph of this Section, the original version of automatic kriging does not perform well with the presence of close points in the design. Adaptive nugget is brought in to resolve this issue and we can check how well it performs now. As shown in Figure 2.11, the performance of LK predictor for design with close points greatly improves after incorporating adaptive nugget. A formal simulation study is also conducted and the result is shown in Figure 2.12. We adopt the one-dimensional function with 100 pairs of close points (mutual distance$< 10^{-6}$). One can see that both OK and LK fail in this case, but LK with adaptive nugget clearly outperforms IDW.

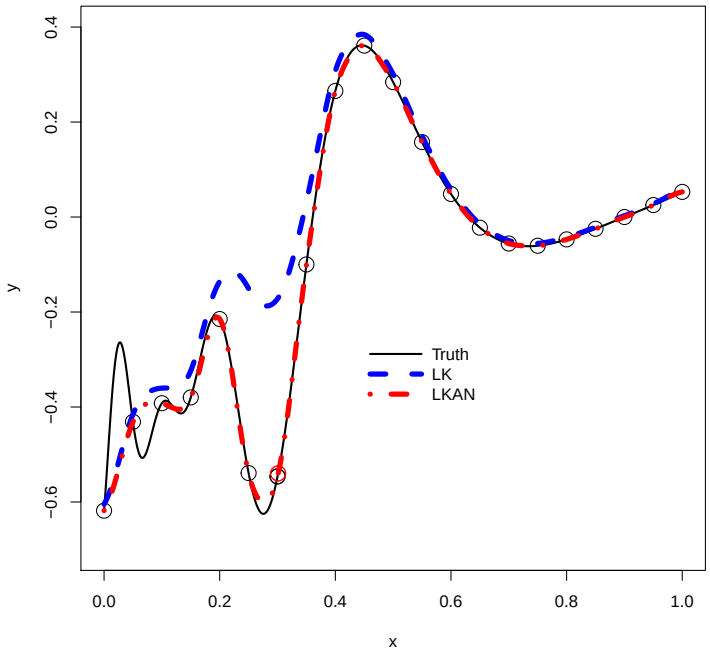

Figure 2.11: The comparison of LK and LKAN with one pair of close points using 1d function.

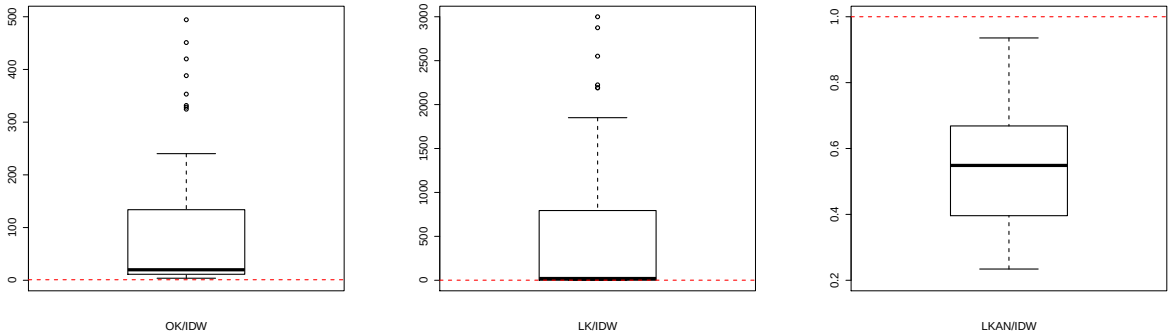

Figure 2.12: Comparing the MSPE of different methods for 1d function with 100 pairs of close points. Left panel: IDW, OK and LK. Right panel: LKAN versus IDW.

The LKAN predictor actually also works well for non-stationary designs. For the one-dimensional function, 75 points are generated from $U(0, 0.5)$ and 25 points are generated from $U(0.5, 1)$. Figure 2.13 shows that LK performs better than IDW and OK, and LKAN with q=0.99 performs even better. The results from Franke function with 72 points from

$U(0, 0.5)^2$ and 28 points from $U(0, 1)^2$, as well as Borehole 8d function with 150 points from $U(0.05, 0.95)^8$ and 150 points from $U(0, 1)^8$ also show similar pattern (Figure 2.14).

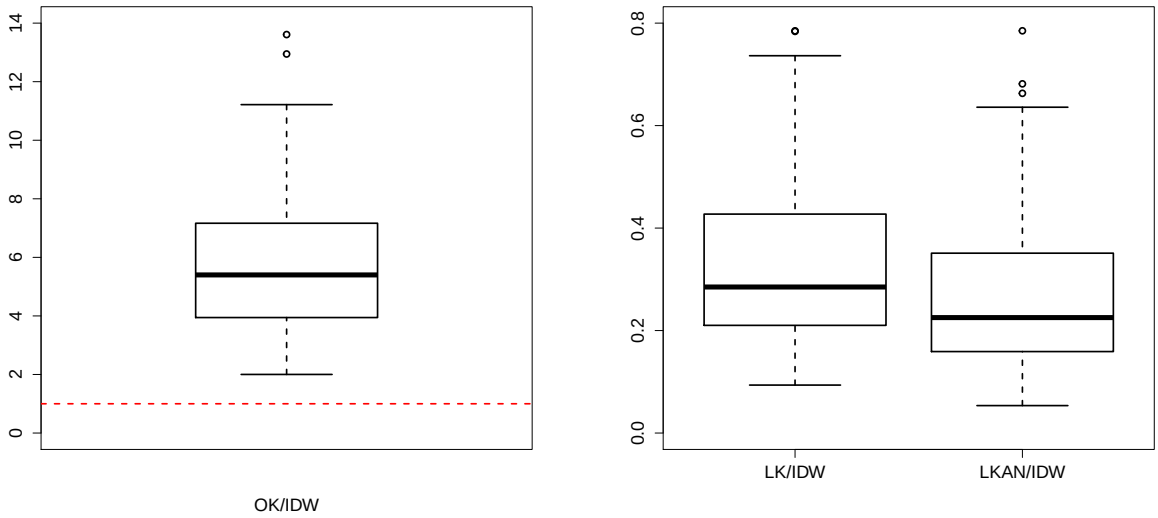

Figure 2.13: Comparing the MSPE of IDW, OK, LK and LKAN for 1d non-uniform design. Left panel: LK versus IDW and OK. Right panel: LKAN versus LK.

In light of sequential usage of design points, we can invert the correlation matrix iteratively using

$$(\boldsymbol{R}_{k+1} + \boldsymbol{\Lambda}_{k+1})^{-1} =$$
$$\begin{bmatrix} (\boldsymbol{R}_k + \boldsymbol{\Lambda}_k)^{-1} + (\boldsymbol{R}_k + \boldsymbol{\Lambda}_k)^{-1}\boldsymbol{r}_k(\boldsymbol{x}_{k+1})\boldsymbol{r}_k^T(\boldsymbol{x}_{k+1})(\boldsymbol{R}_k + \boldsymbol{\Lambda}_k)^{-1}/m & -(\boldsymbol{R}_k + \boldsymbol{\Lambda}_k)^{-1}\boldsymbol{r}_k(\boldsymbol{x}_{k+1})/m \\ -\boldsymbol{r}_k^T(\boldsymbol{x}_{k+1})(\boldsymbol{R}_k + \boldsymbol{\Lambda}_k)^{-1}/m & 1/m \end{bmatrix}, \tag{2.8}$$

where $m = 1 + \lambda_{k+1} - \boldsymbol{r}_k^T(\boldsymbol{x}_{k+1})(\boldsymbol{R}_k + \boldsymbol{\Lambda}_k)^{-1}\boldsymbol{r}_k(\boldsymbol{x}_{k+1})$. The plot of m under different approaches of computing adaptive nugget is displayed in the right panel of Figure 2.10.

One may be concerned with the rounding errors resulted from iterative computation of the inverse of correlation matrix. When m in the formula above is relatively small, the matrix inversion is subject to large numerical errors. We consider the following example

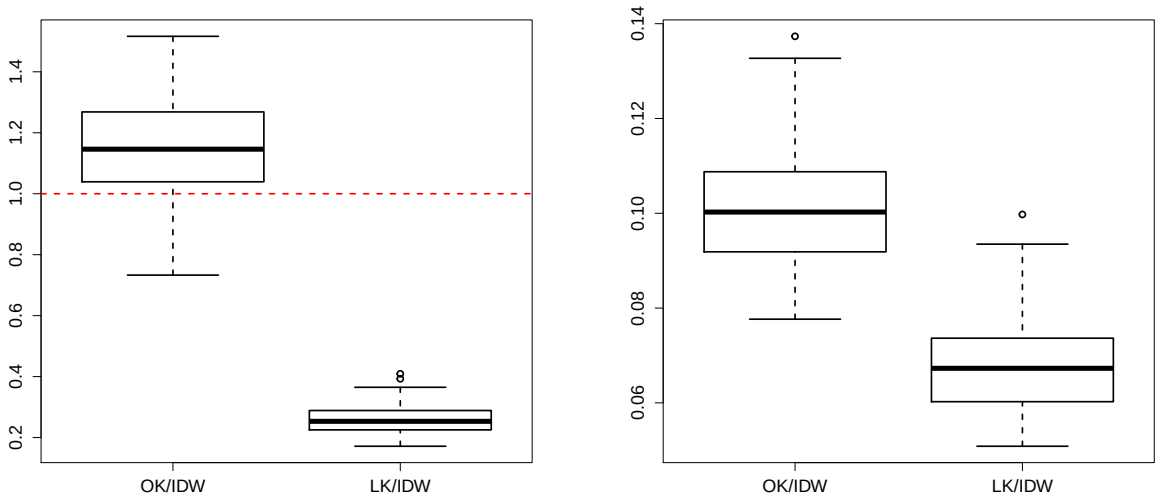

Figure 2.14: Comparing the MSPE of IDW, OK, LK and LKAN for non-uniform design. Left panel: LK versus IDW and OK for 2d Franke function. Right panel: LK versus IDW and OK for 8d Borehole function.

to check on this issue. Suppose there are n equally spaced points in [0,1] and the correlation function of $R(x, w) = \exp(-\theta|x - w|)$ is used. Then, the correlation matrix can be computed as

$$\boldsymbol{R} = \begin{bmatrix} 1 & \rho & \rho^2 & ... & \rho^{n-1} \\ \rho & 1 & \rho & ... & \rho^{n-2} \\ & & & ... & \\ \rho^{n-1} & \rho^{n-2} & & ... & 1 \end{bmatrix},$$

and in this case the true inverse matrix is

$$\boldsymbol{R}^{-1} = \frac{1}{1-\rho^2} \begin{bmatrix} 1 & -\rho & 0 & ... & 0 \\ -\rho & 1+\rho^2 & -\rho & ... & 0 \\ & & ... & & \\ 0 & 0 & ... & -\rho & 1 \end{bmatrix},$$

where $\rho = \exp(-\frac{\theta}{n-1})$. Let $\boldsymbol{R}^{-1}_{solve}$ denote the inverse obtained from standard *solve* function

in R and $\boldsymbol{R}_{iter}^{-1}$ denote the inverse obtained from iterative approach described in this section. We fix $\theta = 1$, and compute the infinity norm $||\cdot||_\infty$ and Frobenius norm $||\cdot||_F$ of the error matrices $\boldsymbol{R}_{solve}^{-1} - \boldsymbol{R}^{-1}$ and $\boldsymbol{R}_{iter}^{-1} - \boldsymbol{R}^{-1}$ with varying design size n. Note that θ used in this example should be much larger than 1 in reality, but we choose this value to amplify the numerical errors. As we can see from Figure 2.15, by reducing the computation through iterative approach, it indeed incurs larger rounding errors than the benchmark method. The magnitude of errors is acceptable with moderately large datasets, but for very large dataset, we need to adopt more stable computational method to control the rounding error.

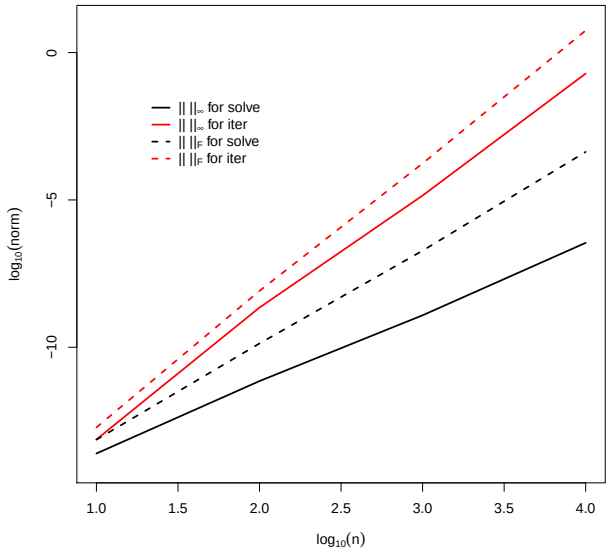

Figure 2.15: Rounding errors of different approaches for computing matrix inverse with varying design size.

2.4 Adaptive Kernel

The traditional Gaussian process models use the same kernel/correlation function across all data points. However, non-constant kernels can be useful to handle unstructured data and non-stationary processes. For example, we can have an unequally-spaced design as shown in the left panel of Figure 2.16. More design points are placed in the left region of the function since this part is more volatile. For this example, it is more reasonable to

have narrower kernels in the left part and wider ones in the right part. However, if we just proceed with the usual limit kriging, the kernel width will be constant as shown in the right panel of Figure 2.16.

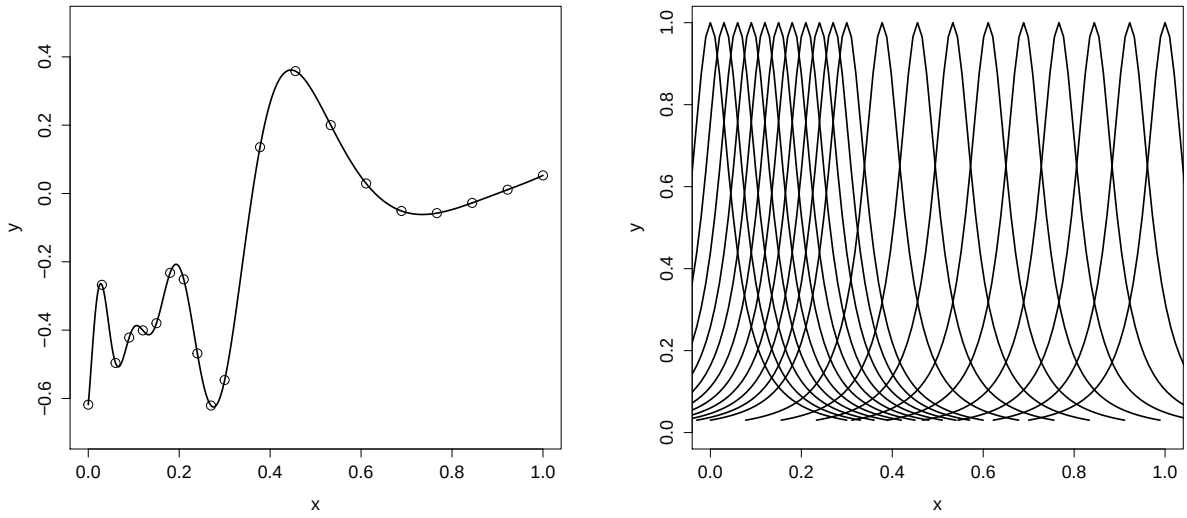

Figure 2.16: Motivating example of adaptive kernel. Left panel: An unequally-spaced design. Right panel: Traditional approach with constant kernel width.

Inspired by the idea of enabling adaptive nugget by sequentially adding design points, we consider further allowing for adaptive kernel in automatic kriging. That is, for each design point, it can have its own kernel function to better represent the dynamics of underlying model. The kernel function for design point $\boldsymbol{x}_i$ is defined as

$$R(\boldsymbol{x}-\boldsymbol{x}_i,\boldsymbol{\theta}_i)=\frac{1}{1+\sum_{j=1}^{p}(\frac{x_j-x_{ij}}{\theta_{ij}})^2}, i=1,...,n,$$

where $\boldsymbol{\theta}_i=(\theta_{i1},...,\theta_{ip})^T$ are the kernel parameters corresponding to design point $\boldsymbol{x}_i$. Assuming the kernel parameters for the design points are known, the predictor based on adaptive kernel model is

$$\hat{y}(\boldsymbol{x})=\frac{\boldsymbol{r}_n^T(\boldsymbol{x}-\boldsymbol{x}_1,...,\boldsymbol{x}-\boldsymbol{x}_n;\boldsymbol{\theta_1},...,\boldsymbol{\theta_n})(\boldsymbol{R}+\boldsymbol{\Lambda})^{-1}\boldsymbol{y}}{\boldsymbol{r}_n^T(\boldsymbol{x}-\boldsymbol{x}_1,...,\boldsymbol{x}-\boldsymbol{x}_n;\boldsymbol{\theta_1},...,\boldsymbol{\theta_n})(\boldsymbol{R}+\boldsymbol{\Lambda})^{-1}\boldsymbol{1}}, \tag{2.9}$$

where $\boldsymbol{r}_n(\boldsymbol{x}-\boldsymbol{x}_1,...,\boldsymbol{x}-\boldsymbol{x}_n;\boldsymbol{\theta_1},...,\boldsymbol{\theta_n})$ is an n-dimensional vector with ith element $R(\boldsymbol{x}-\boldsymbol{x}_i,\boldsymbol{\theta}_i)$, $\boldsymbol{R}$ is the $n \times n$ kernel matrix with ijth element $R(\boldsymbol{x}_i-\boldsymbol{x}_j,\boldsymbol{\theta}_j), i,j=1,...,n$, and $\mathbf{1}$ is the n-dimensional vector composed of ones. Here $\boldsymbol{\Lambda}=\text{diag}(\lambda_1,...,\lambda_n)$ is the diagonal nugget matrix computed using the adaptive nugget approach introduced in Section 2.3 and the details are shown below.

With such type of adaptive kernel, it becomes easier for us to address the non-stationarity of the underlying process. Nevertheless, we need to estimate correlation parameters at each design point now while only one set of correlation parameters needs to be estimated in traditional Gaussian process model. For the adaptive kernel model, it is infeasible to use maximum likelihood estimation due to the dimensionality of the parameters, especially with the presence of a large number of design points. Therefore, we resort to a data-driven approach for estimation. Similar to the adaptive nugget idea, the design points are sequentially added to the current design according to formula (2.7). To simplify the notations, we use $\boldsymbol{x}_1,...,\boldsymbol{x}_n$ to denote the ordered data. Suppose we have already estimated $\boldsymbol{\theta}_1,...,\boldsymbol{\theta}_k$ and computed $\boldsymbol{\lambda}_1,...,\boldsymbol{\lambda}_k, k=1,...,n-1$. Based on the same idea as estimating fixed correlation parameter in (2.3), $\boldsymbol{\theta}_{k+1}$ corresponding to the $k+1$st added design point $\boldsymbol{x}_{k+1}$ is estimated by

$$\hat{\boldsymbol{\theta}}_{k+1}=D_{k+1},$$

where D_{k+1} is the filling distance of $\boldsymbol{x}_{k+1}$. With such definition of kernel width for each design point, the kernel functions in the earlier example are shown in Figure 2.17.

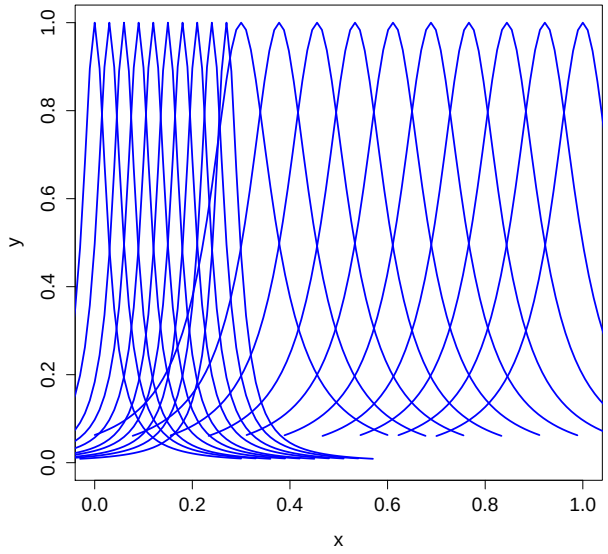

Figure 2.17: Adaptive kernel for the unequally-spaced design example.

Following the adaptive nugget computation described in the last section,

$$\lambda_{k+1} = \frac{1}{(1-q)^3}\{\boldsymbol{r}_k^T(\boldsymbol{x}_{k+1}-\boldsymbol{x}_1, ..., \boldsymbol{x}_{k+1}-\boldsymbol{x}_k; \hat{\boldsymbol{\theta}}_1, ..., \hat{\boldsymbol{\theta}}_k)(\boldsymbol{R}_k+\boldsymbol{\Lambda}_k)^{-1}\boldsymbol{r}_k(\boldsymbol{x}_1-\boldsymbol{x}_{k+1}, ...\boldsymbol{x}_k-\boldsymbol{x}_{k+1}; \hat{\boldsymbol{\theta}}_{k+1})-q\}_+^3,$$

where $\boldsymbol{r}_k(\boldsymbol{x}_1-\boldsymbol{x}_{k+1}, ...\boldsymbol{x}_k-\boldsymbol{x}_{k+1}; \hat{\boldsymbol{\theta}}_{k+1})$ is a k-dimensional vector with ith element $R(\boldsymbol{x}_i - \boldsymbol{x}_{k+1}, \hat{\boldsymbol{\theta}}_{k+1}), i = 1, ..., k$, $\boldsymbol{\Lambda}_k = \text{diag}(\lambda_1, ..., \lambda_k)$ and q is the algorithmic parameter which controls the size of nugget. As is shown in Figure 2.18, limit kriging with adaptive kernel does a much better job in capturing the dynamics of the underlying system than the original LK model.

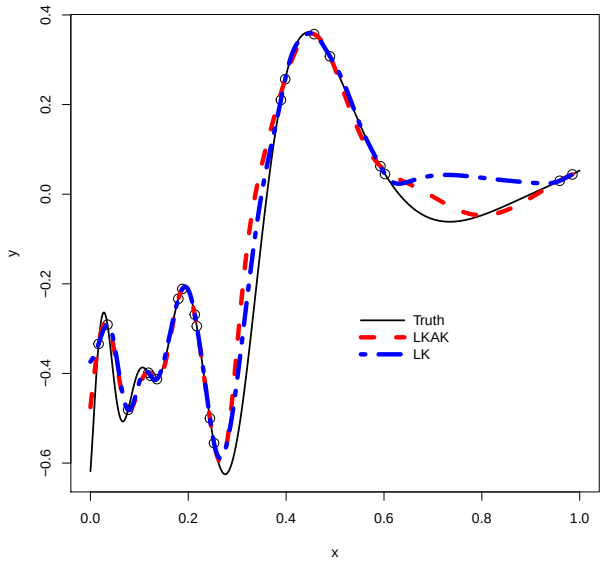

Figure 2.18: The comparison of limit kriging with adaptive kernel with the original version.

2.5 Conclusions

In this Chapter, we propose the estimation-free automatic kriging approach to handle large datasets. The proposed method consistently performs well across various types of design, different design sizes and varying dimensions. In all of the simulated experiments we have conducted, automatic kriging performs better than the inverse distance weighting predictor. Its biggest advantage lies within the ability to deal with large datasets. The automatic kriging approach can save a huge amount of time comparing with traditional methods, and its computation speed is faster than state-of-art methods under appropriate settings. In addition, it can be further improved to handle design with design with close points and non-stationary processes by incorporating adaptive modeling techniques including adaptive nugget and adaptive kernel. In conclusion, the proposed method can serve as a convenient yet powerful tool to model large-scale and unstructured datasets.

2.6 Appendix

2.6.1 On Choosing the Correlation Parameter Estimator for Automatic Kriging

To estimate θ from the filling distances $\boldsymbol{D} = \{D_i, i = 1, ..., n\}$, we can use the generic function

$$\hat{\theta} = cf(\boldsymbol{D}),$$

where reasonable choices of function f can be max, median, mean and quantile defined in (2.3), and c can take 1 or 2. We still use the one-dimensional function and two-dimensional Franke function as our testing functions and generate designs from uniform distribution with different sizes. In Table 2.2, the ratios of MSPE of automatic kriging to that of IDW for 1d function are summarized using maximum and median within 100 simulations. Maximal ratio is used to evaluate the worst-case performance and median ratio is used to provide overall assessment. The ratios of MSPE for Franke 2d function are summarized in Table 2.3. Generally speaking, "max" works well for large n, but is not quite stable with small n. "median" has moderate performance across different design sizes. Using "quantile" balances the strength of both methods. From the definition of the quantile function, it is equivalent to "median" when $n \leq 10p$, and it converges to "max" when $n \to \infty$. In summary, using $f =$ quantile with $c = 1$ is a robust choice across all settings.

Table 2.2: The ratio of MSPE of automatic kriging to IDW under different settings of estimating θ using 1d function(ratios are the smaller the better, numbers in the table are max(median) across 100 simulations)

	n=10		n=20	
	d	2d	d	2d
max	25.7(2.6)	799(42.7)	28.1(1.1)	472(7.9)
median	1.6(0.79)	5.4(1.1)	1.27(0.64)	2.9(0.57)
mean	1.8(0.8)	7.9(1.3)	1.4(0.6)	3.3(0.6)
quantile	1.6(0.79)	5.4(1.1)	2.1(0.57)	12.4(0.77)
	n=50		n=100	
	d	2d	d	2d
max	3.2(0.34)	28.9(0.83)	1.6(0.17)	4.6(0.13)
median	1(0.55)	0.86(0.34)	0.95(0.55)	0.9(0.33)
mean	0.92(0.48)	0.88(0.31)	0.93(0.48)	0.9(0.27)
quantile	1.1(0.31)	1.9(0.31)	0.89(0.21)	1.4(0.13)

Table 2.3: The ratio of MSPE of automatic kriging to IDW under different settings of estimating θ using Franke 2d function

	n=10		n=20	
	d	2d	d	2d
max	2.8(0.45)	29.7(1.13)	0.46(0.15)	37.8(0.69)
median	1.0(0.77)	1.1(0.58)	0.9(0.65)	0.71(0.39)
mean	1.0(0.72)	1.2(0.52)	0.88(0.62)	0.66(0.36)
quantile	1.0(0.77)	1.1(0.58)	0.79(0.52)	0.54(0.27)
	n=50		n=100	
	d	2d	d	2d
max	0.25(0.057)	0.13(0.012)	0.18(0.038)	0.041(0.0052)
median	0.76(0.57)	0.47(0.28)	0.63(0.5)	0.32(0.21)
mean	0.72(0.54)	0.42(0.25)	0.59(0.47)	0.29(0.18)
quantile	0.5(0.3)	0.21(0.096)	0.31(0.19)	0.12(0.043)

CHAPTER 3
ENHANCING AUTOMATIC KRIGING WITH ADAPTIVE MODELING METHODS

This Chapter extends the automatic kriging proposed in Chapter 2 by exploiting the sequential nature of the adaptive modeling method. When more computing resources are available, we have the option to make estimates from adaptive nugget and adaptive kernel more accurate. A two-stage version of adaptive nugget predictor is proposed which is shown to outperform the state-of-the-art methods in terms of prediction accuracy. We also propose fast estimation techniques to improve the adaptive kernel predictor. The improved predictor is demonstrated to have enhanced stability and predictive performance over the traditional kriging method according to various simulation studies.

3.1 Introduction

In Chapter 2, we propose the automatic kriging which has greatly reduced computational compared to traditional kriging methods. This computation reduction mainly comes from inverting the correlation matrix between the design points. However, when there is a large number of observations or when the design points are too close to each other, the correlation matrix can be ill-conditioned which causes numerical instability of the kriging predictor. In such cases, we can add adaptive nugget to resolve the issue as introduced in the last Chapter. Recall that there is a tuning parameter q in computing the adaptive nugget. When q is small, the adaptive nugget tends to be large and will lead to over-smoothed predictor. On the contrary, if q is very close to 1, the nugget added may be too small and the instability issue cannot be completely resolved.

In this chapter, we propose a multi-stage approach to improve the adaptive nugget estimation when more computing resources are available. Iterative approach of building a

multi-stage interpolator is also discussed in [67], [44], [68] among others. [69] proposes the composition of two Gaussian processes which can also be viewed as a two-stage predictor. Based on the sequential nature of adaptive nugget approach, one can freely stop adding design points after collecting enough, to balance the speed and accuracy of the predictor.

In the last chapter, we also proposed the usage of adaptive kernel to model non-stationary processes. Nonetheless, our previous focus was to save the computational cost so it did not make use of the response data but just the design itself. Again, if computation resources are abundant, we can estimate the adaptive correlation parameters in a more cautious manner so as to enhance the performance of our predictor. Since the adaptive kernel approach sequentially adds points to the design, we have the flexibility to choose when to devote more computation resource and when to devote less.

This chapter is organized as follows: Section 3.2 introduces the method of constructing an interpolator through multi-stage adaptive nugget modeling, and then describes the specific way to build a two-stage interpolator in detail. We illustrate how to improve the estimation of adaptive kernel with enough computation power in Section 3.3. The conclusions are drawn in Section 3.4.

3.2 Enhancing Adaptive Nugget Approach

Again, suppose we adopt the Gaussian process model to represent the underlying true process

$$y(\boldsymbol{x}) = \mu + \delta(\boldsymbol{x}), \tag{3.1}$$

where μ is the unknown mean parameter, $\delta(\boldsymbol{x})$ follows the Gaussian process $GP(0, \tau^2 R(\cdot,\cdot))$, τ^2 is the variance of the Gaussian process and $R(\cdot,\cdot)$ denotes the Gaussian correlation function given by

$$R(\boldsymbol{x}, \boldsymbol{x'}) = \exp\{-\sum_{j=1}^{p} \theta_j (x_j - x'_j)^2\},$$

where $\boldsymbol{x} = (x_1, x_2, ..., x_p)^T$ is a p-dimensional vector, and $\boldsymbol{\theta} = (\theta_1, \theta_2, ..., \theta_p)^T$ are the correlation parameters. In this chapter, since more accurate estimates are available and the predictors are less likely to suffer from parameter misspecification, we shall switch to use the ordinary kriging with Gaussian correlation function instead of limit kriging with inverse multiquadric correlation. Suppose we have n standardized design points $\boldsymbol{X} = \{\boldsymbol{x}_i \in [0,1]^p, i = 1, ..., n\}$ and the corresponding responses are $\boldsymbol{y} = (y_1, ..., y_n)^T$. The ordinary kriging predictor based on (3.1) can be written as

$$\hat{y}(\boldsymbol{x}) = \hat{\mu} + \boldsymbol{r}^T(\boldsymbol{x})\boldsymbol{R}^{-1}(\boldsymbol{y} - \hat{\mu}\mathbf{1}), \tag{3.2}$$

where $\hat{\mu} = (\mathbf{1}^T\boldsymbol{R}^{-1}\boldsymbol{y})/(\mathbf{1}^T\boldsymbol{R}^{-1}\mathbf{1})$, $\boldsymbol{R}$ is the $n \times n$ correlation matrix with ijth element $R(\boldsymbol{x}_i, \boldsymbol{x}_j)$, $\boldsymbol{r}(\boldsymbol{x})$ is a n-dimensional vector with ith element $R(\boldsymbol{x}_i, \boldsymbol{x})$ and $\mathbf{1}$ is the n-dimensional vector composed of ones.

3.2.1 Ordinary Kriging-based Interpolator from Infinite Stages

The adaptive nugget approach is proposed in the last chapter to resolve the instability issue in unstructured data. Using the proposed method, the nugget of the next point $\boldsymbol{x}_{k+1}, k = 1, ..., n-1$ is computed as

$$\lambda_{k+1} = \frac{1}{(1-q)^3}\{\boldsymbol{r}_k^T(\boldsymbol{x}_{k+1})(\boldsymbol{R}_k + \boldsymbol{\Lambda}_k)^{-1}\boldsymbol{r}_k(\boldsymbol{x}_{k+1}) - q\}_+^3, \tag{3.3}$$

where $\boldsymbol{r}_k(\boldsymbol{x}_{k+1})$ is a k-dimensional vector with ith element $R(\boldsymbol{x}_i, \boldsymbol{x}_{k+1}), i = 1, ..., k$, $\boldsymbol{\Lambda}_k$ is the diagonal nugget matrix with diagonal elements $\lambda_1, ..., \lambda_k$ and the algorithmic parameter q is to control the size of nugget. The ordinary kriging predictor with adaptive nugget can be written as

$$\hat{y}(\boldsymbol{x}) = \hat{\mu} + \boldsymbol{r}^T(\boldsymbol{x})(\boldsymbol{R} + \boldsymbol{\Lambda})^{-1}(\boldsymbol{y} - \hat{\mu}\mathbf{1}), \tag{3.4}$$

where $\hat{\mu} = \{\mathbf{1}^T\big(\boldsymbol{R} + \boldsymbol{\Lambda}\big)^{-1}\boldsymbol{y}\}/\{\mathbf{1}^T\big(\boldsymbol{R} + \boldsymbol{\Lambda}\big)^{-1}\mathbf{1}\}$. When $\boldsymbol{\Lambda}$ is nonzero, (3.4) is not an interpolator. If a very large q is chosen, the instability issue may not be completely solved. On the other hand, if q is too close to 0, it may lead to over-smoothed predictor. To have a better understanding of the effect of nugget on the predictor, we derive the following relationship.

Theorem 3.2.1. *Let* $\hat{y}_k(\boldsymbol{x}) = \mu + \boldsymbol{r}_k^T(\boldsymbol{x})(\boldsymbol{R}_k + \boldsymbol{\Lambda}_k)^{-1}(\boldsymbol{y}_k - \mu\mathbf{1}_k), k = 1, ..., n.$ *Then*

$$\hat{y}_k(\boldsymbol{x}) - \hat{y}_{k-1}(\boldsymbol{x}) = \frac{1}{m}\{\boldsymbol{r}_{k-1}^T(\boldsymbol{x})(\boldsymbol{R}_{k-1} + \boldsymbol{\Lambda}_{k-1})^{-1}\boldsymbol{r}_{k-1}(\boldsymbol{x}_k) - R(\boldsymbol{x}, \boldsymbol{x}_k)\}\{\hat{y}_{k-1}(\boldsymbol{x}_k) - y_k\},$$

where $m = 1 + \lambda_k - \boldsymbol{r}_{k-1}^T(\boldsymbol{x}_k)(\boldsymbol{R}_{k-1} + \boldsymbol{\Lambda}_{k-1})^{-1}\boldsymbol{r}_{k-1}(\boldsymbol{x}_k)$.

Proof. By partitioning the vectors and matrices into blocks,

$$\hat{y}_k(\boldsymbol{x}) = \mu + \begin{bmatrix} \boldsymbol{r}_k^T(\boldsymbol{x}) & R(\boldsymbol{x}, \boldsymbol{x}_k) \end{bmatrix} (\boldsymbol{R}_k + \boldsymbol{\Lambda}_k)^{-1} \begin{bmatrix} \boldsymbol{y}_{k-1} - \mu\mathbf{1}_{k-1} \\ y_k - \mu \end{bmatrix}.$$

After plugging in formula (2.8) and applying linear algebra, the equation can be proved.

□

Based on Theorem 3.2.1, one can immediately see that when $\lambda_k = \boldsymbol{r}_{k-1}^T(\boldsymbol{x}_k)(\boldsymbol{R}_{k-1} + \boldsymbol{\Lambda}_{k-1})^{-1}\boldsymbol{r}_{k-1}(\boldsymbol{x}_k)$,

$$\hat{y}_k(\boldsymbol{x}_k) = \begin{cases} \hat{y}_{k-1}(\boldsymbol{x}_k), & \lambda_k = 1, \\ y_k, & \lambda_k = 0. \end{cases}$$

Thus when λ_k gets close to 1, the predictor hardly changes compared with that in the previous step. On the other hand, the predictor becomes an interpolator when λ_k goes to 0.

To avoid the burden of choosing the right tuning parameter q, we propose the multi-stage version of adaptive nugget predictor. Now we drop the tuning parameter q and switch back to the original way of computing adaptive nugget by fixing the determinant

$$\lambda_{k+1} = \boldsymbol{r}_k^T(\boldsymbol{x}_{k+1})(\boldsymbol{R}_k + \boldsymbol{\Lambda}_k)^{-1}\boldsymbol{r}_k(\boldsymbol{x}_{k+1}). \tag{3.5}$$

Let $\boldsymbol{e}^{(0)}$ denote the residuals $\boldsymbol{y} - \hat{\boldsymbol{y}}$ where $\hat{\boldsymbol{y}} = (\hat{y}(\boldsymbol{x}_1), ..., \hat{y}(\boldsymbol{x}_n))^T$. Then we replace $\boldsymbol{y}$ in (3.4) with $\boldsymbol{e}^{(0)}$ and obtain the predictor $\hat{y}^{(1)}(\boldsymbol{x})$ as well as the residuals $\boldsymbol{e}^{(1)} = \boldsymbol{e}^{(0)} - \hat{\boldsymbol{y}}^{(1)}$. By repeatedly fitting (3.4) to the residuals, we can obtain a sequence of predictors $\{\hat{y}^{(s)}(\boldsymbol{x})\}$ and their corresponding residuals $\{\boldsymbol{e}^{(s)}\}, s = 0, 1, 2, ..., +\infty$. Assuming the set of correlation parameters in stage s is $\boldsymbol{\theta}^{(s)}$, it can be shown that the corresponding nugget matrix $\boldsymbol{\Lambda}^{(s)} \to \mathbf{0}$ as $s \to +\infty$, if we restrict $\boldsymbol{\theta}^{(s)}$ to be monotonically increasing. Based on this, we can easily show

$$\sum_{s=0}^{+\infty} \hat{y}^{(s)}(\boldsymbol{x}_i) = y_i, i = 1, ..., n,$$

implying that the sum of the sequence of predictors is an interpolator. Similar strategy of iterative construction of a predictor is also used in [44].

3.2.2 Stable Ordinary Kriging-based Interpolator from Two Stages

In practice, it is unrealistic to fit Gaussian process model for infinite times. In this section, a two-stage method is proposed to build a stable interpolator which demonstrates robust performance across different designs. Similar idea can be found in [69], which uses the composition of two Gaussian process to increase the stability and accuracy of prediction.

Stage 1. Ordinary kriging with adaptive nugget. In the first stage, we still fit the ordinary kriging predictor (3.4) to the data. In Chapter 2, we decide the order of sequentially adding points according to (2.7), with the idea of picking the next point with minimal nugget. Using Gaussian model in (3.1), the predictive variance of a new observation can be computed as

$$var(y(\boldsymbol{x})|\boldsymbol{y}) = \tau^2(1 - \boldsymbol{r}^T(\boldsymbol{x})(\boldsymbol{R} + \boldsymbol{\Lambda})^{-1}\boldsymbol{r}(\boldsymbol{x})).$$

Thus, choosing the next point with minimal nugget is equivalent to choosing that with the largest variance. Such type of sequential design idea can also be found in [70], [71], [72]. This approach is purely based on the input variables without using their responses.

Now, with more computational budget, we have the option of choosing the next point with the worst fit to the current model. The first point $\boldsymbol{x}_{(1)}$ is still chosen to be $\boldsymbol{x}_{(1)} = arg\max_{\boldsymbol{x}_i}\{|y(\boldsymbol{x}_i) - \bar{y}|, i = 1, ..., n\}$ where $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$. Suppose we already have k ordered points $\boldsymbol{x}_{(1)}, ..., \boldsymbol{x}_{(k)}$, $k = 1, ..., n-1$ and have computed their corresponding nuggets according to (3.5). Then we fit an ordinary kriging predictor to the ordered data by

$$\hat{y}_k(\boldsymbol{x}) = \bar{y} + \boldsymbol{r}_k^T(\boldsymbol{x})(\boldsymbol{R}_k + \boldsymbol{\Lambda}_k)^{-1}(\boldsymbol{y}_k - \bar{y}\mathbf{1}), \tag{3.6}$$

where $\boldsymbol{r}_k, \boldsymbol{R}_k, \boldsymbol{\Lambda}_k$ are defined the same as the previous section and $\boldsymbol{y}_k = (y_{(1)}, ..., y_{(k)})^T$. The next point $\boldsymbol{x}_{(k+1)}$ is decided based on

$$\boldsymbol{x}_{(k+1)} = arg\max_{\boldsymbol{x}_i}\{|\hat{y}_k(\boldsymbol{x}_i) - y_i|, \boldsymbol{x}_i \notin \boldsymbol{X}_k, i = 1, ..., n\},$$

where $\boldsymbol{X}_k = \{\boldsymbol{x}_{(1)}, ..., \boldsymbol{x}_{(k)}\}$. This enables us to iteratively obtain the fully ordered data $\boldsymbol{x}_{(1)}, ..., \boldsymbol{x}_{(n)}$ and the nugget matrix $\boldsymbol{\Lambda}$. As a matter of fact, the computational complexity of the two ways to compute nugget is the same.

The parameter $\mu, \tau^2, \boldsymbol{\theta}$ in model (3.1) are unknown and need to be estimated. We can still estimate them based on the distance-based approach as in Chapter 2, but when there are enough budget for computing, we have the option to improve the distance-based estimates to more accurate model-based estimates. The MLE's of μ and τ^2 can both be expressed as functions of $\boldsymbol{\theta}$, and the profile likelihood of $\boldsymbol{\theta}$ can be written as

$$L(\boldsymbol{\theta}) = \frac{\exp\{-n/2\}}{(2\pi\hat{\tau}^2)^{n/2}|\boldsymbol{R} + \boldsymbol{\Lambda}|^{1/2}},$$

where $\boldsymbol{R}, \boldsymbol{\Lambda}$ are functions of $\boldsymbol{\theta}$, $\hat{\tau}^2 = \frac{1}{n}(\boldsymbol{y} - \hat{\mu}\mathbf{1})^T(\boldsymbol{R} + \boldsymbol{\Lambda})^{-1}(\boldsymbol{y} - \hat{\mu}\mathbf{1})$, and $\hat{\mu} = \{\mathbf{1}^T(\boldsymbol{R} + \boldsymbol{\Lambda})^{-1}\boldsymbol{y}\}/\{\mathbf{1}^T(\boldsymbol{R} + \boldsymbol{\Lambda})^{-1}\mathbf{1}\}$. Maximizing the likelihood is equivalent to minimize the deviance

$$-2\log L(\boldsymbol{\theta}) = n\log\hat{\tau}^2 + \log|\boldsymbol{R} + \boldsymbol{\Lambda}| + n + n\log 2\pi.$$

Nevertheless, the MLE does not perform well in many cases, especially when the sample size is small. So we adopt an estimation criterion which resembles the one proposed in [73], aiming at jointly optimization of the likelihood and the variance of kriging prediction

$$C_{DET}(\boldsymbol{\theta}) = (n+1)\log\hat{\tau}^2 + \log|\boldsymbol{R}+\boldsymbol{\Lambda}| + \max_{\boldsymbol{x}\notin\boldsymbol{X}} \log\{1-\boldsymbol{r}^T(\boldsymbol{x})(\boldsymbol{R}+\boldsymbol{\Lambda})^{-1}\boldsymbol{r}(\boldsymbol{x})\}.$$

We make a little modification to the criterion above by replacing the maximization with integration, i.e.,

$$\tilde{C}_{DET}(\boldsymbol{\theta}) = (n+1)\log\hat{\tau}^2 + \log|\boldsymbol{R}+\boldsymbol{\Lambda}| + \log\int_{[0,1]^p} \{1-\boldsymbol{r}^T(\boldsymbol{x})(\boldsymbol{R}+\boldsymbol{\Lambda})^{-1}\boldsymbol{r}(\boldsymbol{x})\}d\boldsymbol{x}. \quad (3.7)$$

The modified criterion intends to minimize the overall prediction variance while the original one just minimizes the worst-case variance. The explicit expression of the integral in (3.7) is readily available per our usage of Gaussian correlation function.

Theorem 3.2.2. *The modified criterion can be expressed as*

$$\tilde{C}_{DET}(\boldsymbol{\theta}) = (n+1)\log\hat{\tau}^2 + \log|\boldsymbol{R}+\boldsymbol{\Lambda}| + \log\{1-tr[(\boldsymbol{R}+\boldsymbol{\Lambda})^{-1}\boldsymbol{S}]\},$$

where S denotes the $n\times n$ matrix $\int_{[0,1]^p}\boldsymbol{r}(\boldsymbol{x})\boldsymbol{r}^T(\boldsymbol{x})d\boldsymbol{x}$ with $ijth$ element S_{ij}.

Proof. Based on results from linear algebra, (3.7) can be simplified as

$$\tilde{C}_{DET}(\boldsymbol{\theta}) = (n+1)\log\hat{\tau}^2 + \log|\boldsymbol{R}+\boldsymbol{\Lambda}| + \log\{1-tr[(\boldsymbol{R}+\boldsymbol{\Lambda})^{-1}\int_{[0,1]^p}\boldsymbol{r}(\boldsymbol{x})\boldsymbol{r}^T(\boldsymbol{x})d\boldsymbol{x}]\}.$$

Let $S_{ij} = \int_{[0,1]^p} s_{ij}(\boldsymbol{x})d\boldsymbol{x}$, where

$$\begin{aligned} s_{ij}(\boldsymbol{x}) &= \exp\{-(\boldsymbol{x}-\boldsymbol{x}_i)^T\boldsymbol{\Theta}(\boldsymbol{x}-\boldsymbol{x}_i) - (\boldsymbol{x}-\boldsymbol{x}_j)^T\boldsymbol{\Theta}(\boldsymbol{x}-\boldsymbol{x}_j)\} \\ &= \exp\{-\frac{1}{2}[\boldsymbol{x}-\frac{\boldsymbol{x}_i+\boldsymbol{x}_j}{2}]^T 4\boldsymbol{\Theta}[\boldsymbol{x}-\frac{\boldsymbol{x}_i+\boldsymbol{x}_j}{2}] - \frac{1}{2}(\boldsymbol{x}_i-\boldsymbol{x}_j)^T\boldsymbol{\Theta}(\boldsymbol{x}_i-\boldsymbol{x}_j)\}. \end{aligned}$$

Here, $\boldsymbol{\Theta}$ is the diagonal matrix with diagonal elements being $\boldsymbol{\theta}$. According to the multivariate normal density function, we have

$$S_{ij} = (\pi/2)^{p/2}|\boldsymbol{\Theta}|^{-1/2}\exp\{-\frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{x}_j)^T\boldsymbol{\Theta}(\boldsymbol{x}_i - \boldsymbol{x}_j)\}\int_{[0,1]^p}\phi(\boldsymbol{x}; \frac{\boldsymbol{x}_i + \boldsymbol{x}_j}{2}, \frac{1}{4}\boldsymbol{\Theta}^{-1})d\boldsymbol{x},$$

where $\phi(\boldsymbol{x}; \frac{\boldsymbol{x}_i+\boldsymbol{x}_j}{2}, \frac{1}{4}\boldsymbol{\Theta}^{-1})$ is the normal density function with mean $\frac{\boldsymbol{x}_i+\boldsymbol{x}_j}{2}$ and covariance matrix $\frac{1}{4}\boldsymbol{\Theta}^{-1}$, and S_{ij} can be easily evaluated through the cumulative distribution function of multivariate normal distribution. Thus the modified criterion can be rewritten as

$$\tilde{C}_{DET}(\boldsymbol{\theta}) = (n+1)\log\hat{\tau}^2 + \log|\boldsymbol{R} + \boldsymbol{\Lambda}| + \log\{1 - tr[(\boldsymbol{R} + \boldsymbol{\Lambda})^{-1}\boldsymbol{S}]\}.$$

Note that $|\boldsymbol{R} + \boldsymbol{\Lambda}| = 1$ by adding the adaptive nugget using (3.5), which further simplifies our criterion to

$$\tilde{C}_{DET}(\boldsymbol{\theta}) = (n+1)\log\hat{\tau}^2 + \log\{1 - tr[(\boldsymbol{R} + \boldsymbol{\Lambda})^{-1}\boldsymbol{S}]\}.$$

□

We compare the results of two-stage adaptive nugget approch from $\tilde{C}_{DET}$ and MLE criterion using the one-dimensional function with 100 uniformly-distributed design points in Figure 3.1. There are 100 simulation runs in our experiment. The result from MLE estimation is similar to that from *GPfit* library, but the result based on $\tilde{C}_{DET}$ which integrates the optimization of likelihood and mean squared prediction error is much better than the other two.

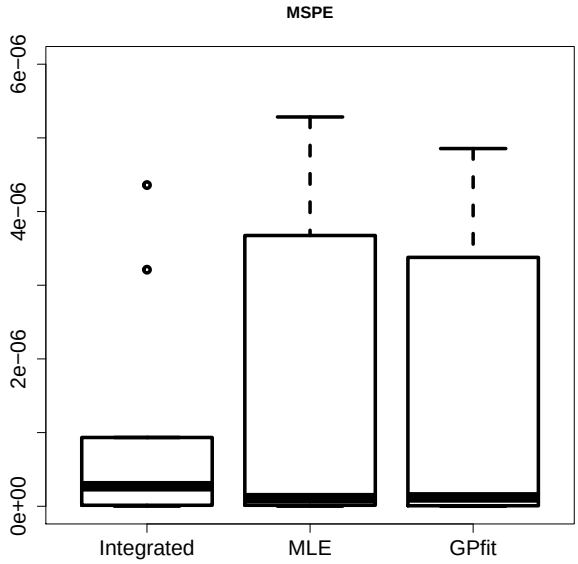

Figure 3.1: The comparison of different estimation approaches using 1d function with 100 design points

It is well known that the correlation parameter $\boldsymbol{\theta}$ in Gaussian process model is not easy to optimize. One needs to run the optimization procedure using multiple starting points with carefully-chosen lower and upper bounds. According to our own experiences, the bounds based on the distances among design points work effectively for most of the time. Let $\bar{d}$ denote the harmonic mean of Euclidean distance between every pair of design points. If the aim is to control the correlation within $[r_l, r_u]$, we can specify the bounds for optimization to be $\theta \in [-\log r_u/\bar{d}^2, -\log r_l/\bar{d}^2]$ according to the Gaussian correlation function.

After applying the predictor in (3.4) for the first stage, the fit for function $y = \sin[30(x-0.9)^4]\cos(2x-1.8) + (x-0.9)/2$ with 40 equally-spaced design points is displayed in Figure 3.2. It captures the global trend of the true function but does not work well for the local fluctuations. Hence we shall introduce a second stage of fitting.

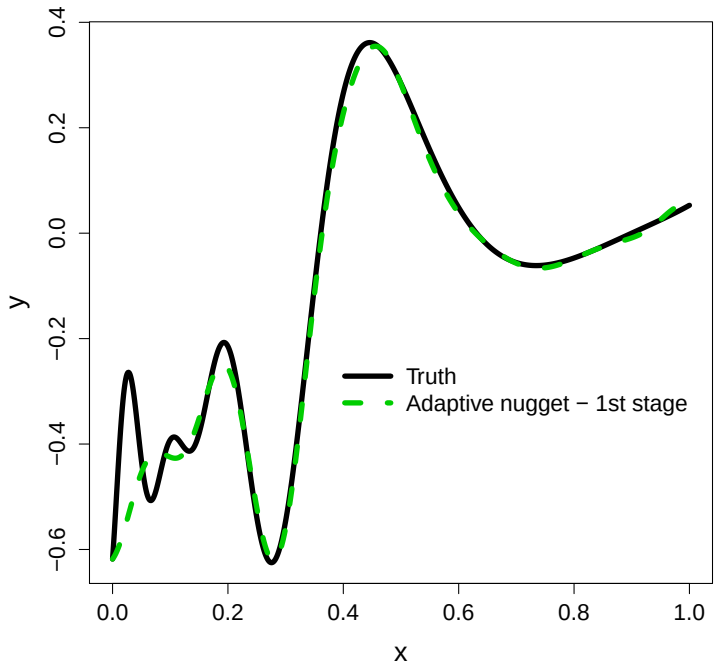

Figure 3.2: The first stage of adaptive nugget predictor with 40 equally spaced design points

Stage 2. Fit ordinary kriging to the residuals without nugget. Let $\hat{\boldsymbol{y}}$ denote the fitted values from stage 1 using (3.4) and $\boldsymbol{e} = \boldsymbol{y} - \hat{\boldsymbol{y}}$ denote the residuals. An ordinary kriging without nugget as in (3.2) is fitted to $\boldsymbol{e}$, which yields an interpolator $\hat{e}(\boldsymbol{x})$. As a consequence, the two-stage predictor $\hat{y}_{AN}(\boldsymbol{x}) = \hat{y}(\boldsymbol{x}) + \hat{e}(\boldsymbol{x})$ interpolates the original data $\boldsymbol{y}$. Let $\hat{\boldsymbol{\theta}}_1$ denote the optimized correlation parameter from stage 1. The constraint $\boldsymbol{\theta}_2 > \hat{\boldsymbol{\theta}}_1$ is added when optimizing $\boldsymbol{\theta}_2$ in the second step, as the first stage captures the global trend and the second stage depicts local details. Figure 3.3 shows the performance of our predictor after applying the second-stage fit to the residuals. Now, the predicted response almost overlies the true function. Similar story can be found in Figure 3.4 for two-stage fit of Franke 2d function using 100 points from Latin hypercube design.

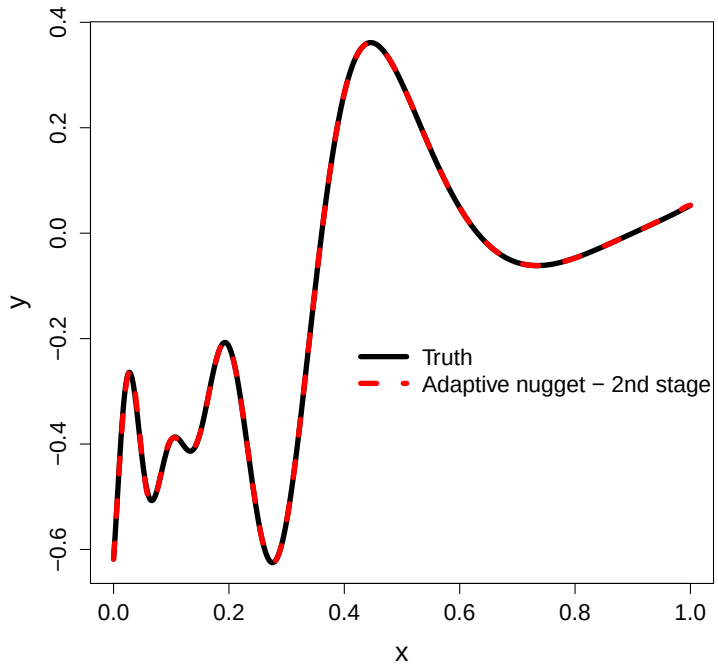

Figure 3.3: Two-stage adaptive nugget predictor with 40 equally spaced design points

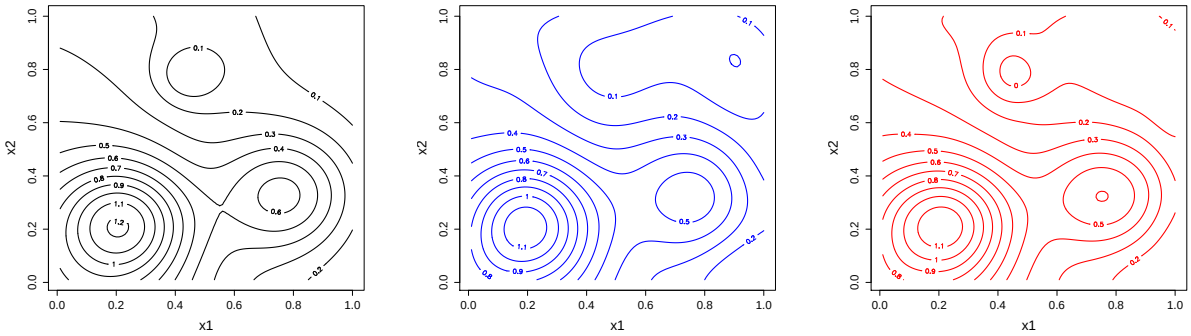

Figure 3.4: Two-stage fitting with adaptive nugget using Franke 2d function with 100 points from Latin hypercube design. Left panel: The contour plot of the Franke function. Middle panel: Plot of the fit after the first stage. Right panel: Plot of the fit after the second stage.

To save the computation effort utilizing the sequential nature of adaptive nugget approach, we can stop adding points when the nugget λ for some design point is larger than a certain threshold value λ_0, since in this case the error is comparable to the variance of Gaussian process and the point does not provide much information. When we sequentially

add nuggets to the design points, the nuggets display an increasing trend. One can determine the design size used in the first stage by adjusting the threshold λ_0. In the second stage, we still fit an ordinary kriging without nugget to the residuals from the smoother in the first stage. Again, we do not need to use all the design points. We can stop picking new points when the current fit is satisfactory, or adopt other sequential design strategies, see [74] among others.

In summary, the two-stage interpolator with adaptive nugget can be computed in the following way

$$\begin{aligned}
\hat{y}(\boldsymbol{x}) &= \hat{\mu} + \boldsymbol{r}^T(\boldsymbol{x})(\boldsymbol{R}+\boldsymbol{\Lambda})^{-1}(\boldsymbol{y}-\hat{\mu}\mathbf{1}),\\
\hat{e}(\boldsymbol{x}) &= \hat{\eta} + \boldsymbol{s}^T(\boldsymbol{x})\boldsymbol{S}^{-1}\{\boldsymbol{e}-\hat{\eta}\mathbf{1}\}\\
&= \hat{\eta} + \boldsymbol{s}^T(\boldsymbol{x})\boldsymbol{S}^{-1}\{\boldsymbol{H}(\boldsymbol{y}-\hat{\mu}\mathbf{1})-\hat{\eta}\mathbf{1}\},\\
\hat{y}_{AN}(\boldsymbol{x}) &= \hat{y}(\boldsymbol{x}) + \hat{e}(\boldsymbol{x}),
\end{aligned}$$

where $\hat{\mu} = \{\mathbf{1}^T(\boldsymbol{R}+\boldsymbol{\Lambda})^{-1}\boldsymbol{y}\}/\{\mathbf{1}^T(\boldsymbol{R}+\boldsymbol{\Lambda})^{-1}\mathbf{1}\}, \hat{\eta} = \{\mathbf{1}^T\boldsymbol{S}^{-1}\boldsymbol{e}\}/\{\mathbf{1}^T\boldsymbol{S}^{-1}\mathbf{1}\}$, $\boldsymbol{H} = \boldsymbol{I}-\boldsymbol{R}(\boldsymbol{R}+\boldsymbol{\Lambda})^{-1}$, $\boldsymbol{r}(\boldsymbol{x})$ and $\boldsymbol{R}$ correspond to correlation function with $\hat{\boldsymbol{\theta}}_1$, $\boldsymbol{s}(\boldsymbol{x})$ and $\boldsymbol{S}$ correspond to correlation function with $\hat{\boldsymbol{\theta}}_2$. It can be easily verified that $E(\hat{y}(\boldsymbol{x})) = E(\hat{\mu}) = \mu, E(\hat{e}(\boldsymbol{x})) = E(\hat{\eta}) = 0$. Thus $\hat{y}_{AN}(\boldsymbol{x})$ is an unbiased predictor of $y(\boldsymbol{x})$. Now to simplify the computations, assume $\hat{\mu} = \mu, \hat{\eta} = 0$. In fact, incorporating the variance of $\hat{\mu}, \hat{\eta}$ does not add any technical difficulty but just algebraic computations. Then, the mean squared prediction error (MSPE) of the two-stage predictor can be computed as follows

$$\begin{aligned}
\text{MSPE}(\hat{y}_{AN}(\boldsymbol{x})) =& E(\hat{y}_{AN}(\boldsymbol{x})-y(\boldsymbol{x}))^2 = var(\hat{y}(\boldsymbol{x})+\hat{e}(\boldsymbol{x})-y(\boldsymbol{x}))\\
=& var(\hat{y}(\boldsymbol{x})) + var(\hat{e}(\boldsymbol{x})) + var(y(\boldsymbol{x})) + 2cov(\hat{y}(\boldsymbol{x}),\hat{e}(\boldsymbol{x}))\\
&- 2cov(\hat{y}(\boldsymbol{x}),y(\boldsymbol{x})) - 2cov(\hat{e}(\boldsymbol{x}),y(\boldsymbol{x}))\\
=& \tau^2\{1-\boldsymbol{r}^T(\boldsymbol{x})(\boldsymbol{R}+\boldsymbol{\Lambda})^{-1}\boldsymbol{r}(\boldsymbol{x}) + \boldsymbol{s}^T(\boldsymbol{x})\boldsymbol{S}^{-1}\boldsymbol{H}(\boldsymbol{R}+\boldsymbol{\Lambda})\boldsymbol{H}\boldsymbol{S}^{-1}\boldsymbol{s}(\boldsymbol{x})\},
\end{aligned}$$

based on which we can build the confidence interval for $y(\boldsymbol{x})$ from the two-stage adaptive nugget predictor.

Now, we provide simulation results to demonstrate the strength of enhanced version of adaptive nugget predictor. The two-stage AN predictor is compared with the traditional ordinary kriging implemented via the popular R package *GPfit*. In *GPfit*, a fixed nugget is added to the correlation matrix according to the lower bound proposed in [53] to avoid over-smoothing. Suppose the true function is $y = \sin[30(x-0.9)^4]\cos(2x-1.8) + (x-0.9)/2$ and we generate $n = 70$ design points $\boldsymbol{X}$ from uniform distribution $U(0,1)$. The testing data is $n_{test} = 1001$ equally spaced points in $[0,1]$. We repeat the simulations for $N = 100$ times and summarize the mean squared prediction error (MSPE) for both AN and GPfit, which is computed through

$$\text{MSPE} = \frac{1}{n_{test}} \sum_{i=1}^{n_{test}} (\hat{y}(\boldsymbol{x}_i) - y(\boldsymbol{x}_i))^2.$$

To better compare the two approaches, the ratio of their MSPE is computed for each simulation. According to Figure 3.5, adaptive nugget achieves better performance than ordinary kriging implemented via the *GPfit* package, and their difference becomes even larger for non-uniform design as shown in Figure 3.6.

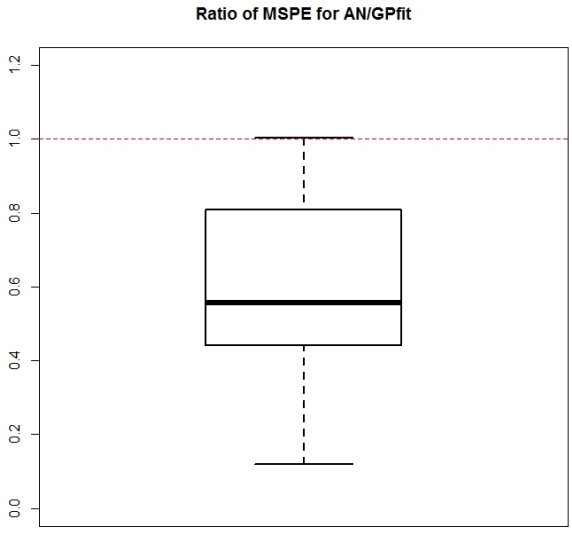

Figure 3.5: The comparison of adaptive nugget predictor and ordinary kriging with 70 design points from $U(0,1)$ using 1d function

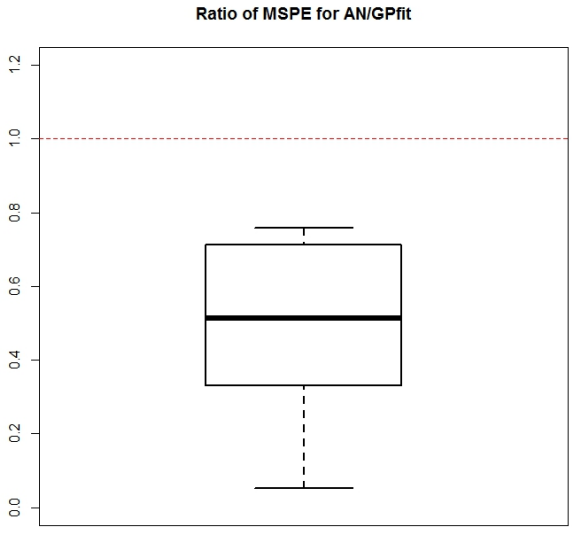

Figure 3.6: The comparison of adaptive nugget predictor and ordinary kriging with 70 points from non-uniform design using 1d function: 56 points from U(0,0.5),14 points from U(0.5,1)

We also try the proposed method on higher dimensional functions. We use the Franke

2d function

$$y = 0.75\exp\{-\frac{(9x_1-2)^2}{4} - \frac{(9x_2-2)^2}{4}\} + 0.75\exp\{-\frac{(9x_1+1)^2}{49} - \frac{9x_2+1}{10}\}$$
$$+ 0.5\exp\{-\frac{(9x_1-7)^2}{4} - \frac{(9x_2-3)^2}{4}\} - 0.2\exp\{-(9x_1-4)^2 - (9x_2-7)^2\}$$

as the underlying true function and generate design points from $U(0,1)^2$. The design size is $n = 70$ and the testing data is a Sobol sequence with $n_{test} = 1000$. Again, we run the simulation 100 times and summarize the ratios of MSPE for adaptive nugget and GPfit. It can be seen from Figure 3.7 and 3.8 that AN still has better overall performance than ordinary kriging.

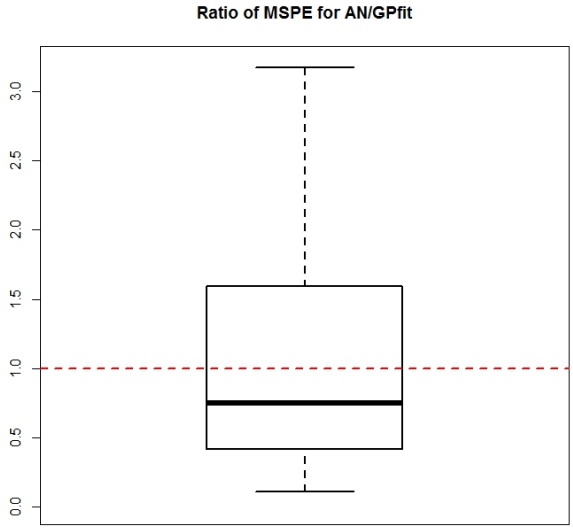

Figure 3.7: The comparison of adaptive nugget predictor and ordinary kriging with 70 design points from $U[0,1]^2$ using Franke 2d function

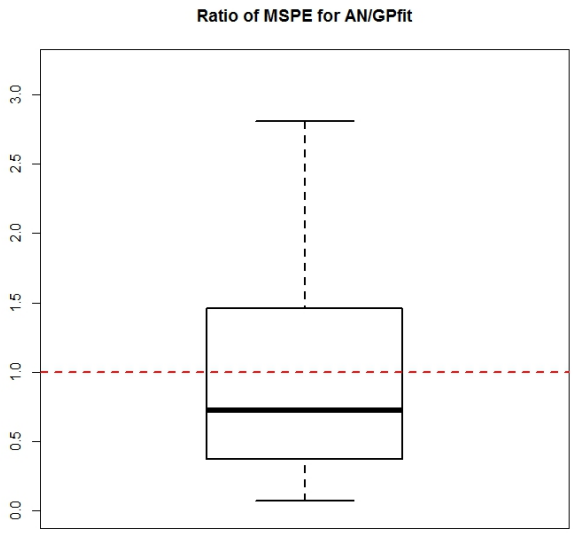

Figure 3.8: The comparison of adaptive nugget predictor and ordinary kriging with 20 design points from $U[0, 0.5]^2$ and 50 design points from $U[0, 1]^2$ using Franke 2d function

Then, we use the Borehole 8d function

$$y = \frac{2\pi T_u(H_u - H_l)}{\ln(r/r_w)\{1 + \frac{2LT_u}{\ln(r/r_w)r_w^2 K_w} + T_u/T_l\}}$$

as the underlying true function and generate design points from $U(0, 1)^8$. The design size is $n = 150$ and the testing data is a Sobol sequence of size 1000. The ratios of MSPE for adaptive nugget and GPfit with 100 simulations are summarized in Figure 3.9, and two-stage adaptive nugget approach is shown to outperform ordinary kriging for higher dimensional examples.

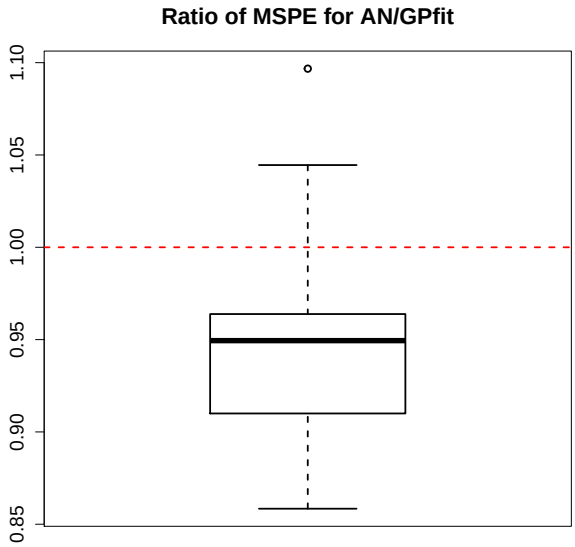

Figure 3.9: The comparison of adaptive nugget predictor and ordinary kriging with 150 design points from $U[0,1]^8$ using Borehole 8d function

3.3 Enhancing Adaptive Kernel Approach

In chapter 2, the adaptive kernel estimates are obtained directly from the design but the response is not used. Given sufficient time and computational resources, we also hope to make use of the response data to make the kernel parameter estimates more accurate. The kernel function for design point $\boldsymbol{x}_i$ is defined as

$$R(\boldsymbol{x}-\boldsymbol{x}_i,\boldsymbol{\theta}_i)=\exp\{-\sum_{j=1}^{p}\theta_{ij}(x_j-x_{ij})^2\},$$

where $\boldsymbol{\theta}_i=(\theta_{i1},...,\theta_{ip})^T$ are the kernel parameters corresponding to design point $\boldsymbol{x}_i$. Suppose we have n design points $\boldsymbol{X}=\{\boldsymbol{x}_i\in\mathbb{R}^p, i=1,...,n\}$ and the corresponding responses are $\boldsymbol{y}=(y_1,...,y_n)^T$. Assuming the kernel parameters for the design points are known, the predictor based on the adaptive kernel model is

$$\hat{y}(\boldsymbol{x})=\hat{\mu}+\boldsymbol{r}_n^T(\boldsymbol{x}-\boldsymbol{x}_1,...,\boldsymbol{x}-\boldsymbol{x}_n;\boldsymbol{\theta}_1,...,\boldsymbol{\theta}_n)(\boldsymbol{R}+\boldsymbol{\Lambda})^{-1}(\boldsymbol{y}-\hat{\mu}\boldsymbol{1}), \tag{3.8}$$

where $\hat{\mu} = \{\mathbf{1}^T\big(\boldsymbol{R}+\boldsymbol{\Lambda}\big)^{-1}\boldsymbol{y}\}/\{\mathbf{1}^T\big(\boldsymbol{R}+\boldsymbol{\Lambda}\big)^{-1}\mathbf{1}\}$, $\boldsymbol{r}_n(\boldsymbol{x}-\boldsymbol{x}_1,...,\boldsymbol{x}-\boldsymbol{x}_n;\boldsymbol{\theta}_1,...,\boldsymbol{\theta}_n)$ is an n-dimensional vector with ith element $R(\boldsymbol{x}-\boldsymbol{x}_i,\boldsymbol{\theta}_i)$, $\boldsymbol{R}$ is the $n\times n$ correlation matrix with ijth element $R(\boldsymbol{x}_i-\boldsymbol{x}_j,\boldsymbol{\theta}_j), i,j=1,...,n$, and $\mathbf{1}$ is the n-dimensional vector composed of ones. Here $\boldsymbol{\Lambda} = \text{diag}(\lambda_1,...,\lambda_n)$ is the diagonal nugget matrix computed using the adaptive nugget approach introduced before.

3.3.1 Improved Estimation of Adaptive Kernel

It becomes much more convenient for us to address the non-stationarity of the underlying process with the aid of adaptive kernel. But now we need to estimate correlation parameters for each design point while one single set of correlation parameters is enough for estimation in traditional Gaussian process model. In the last Chapter, we simply use the filling distance of a design point as an estimate of the correlation parameter, but now we hope to make it more accurate since more computational time is available. For the adaptive kernel model, it is still infeasible to use maximum likelihood estimation due to the dimensionality of the parameters, especially with the presence of a large number of design points. Therefore, we resort to a data-driven approach for estimation. Similar to the adaptive nugget idea, our principle is to add design points sequentially according to the fit in previous step. As shown earlier, we can obtain the ordered data and their corresponding nuggets simultaneously using iterative approach. To simplify the notations, we use $\boldsymbol{x}_1,...,\boldsymbol{x}_n$ to denote the ordered data. The correlation parameters $\boldsymbol{\theta}_k$ corresponding to the kth added design point $\boldsymbol{x}_k$ is estimated according to the local fit.

Let $\mathcal{N}_k$ denote the indices of $\boldsymbol{x}_k$'s neighbors consisting of $4p+1$ points closest to $\boldsymbol{x}_k$. Assume we already obtain estimates $\hat{\boldsymbol{\theta}}_1,...,\hat{\boldsymbol{\theta}}_{k-1}$ for $k-1$ design points. Then we find the correlation parameters of $\boldsymbol{x}_k$ by minimizing

$$E(\boldsymbol{\theta}_k) = \sum_{i\in\mathcal{N}_k}(y_i - \hat{y}_k(\boldsymbol{x}_i;\boldsymbol{\theta}_k))^2. \tag{3.9}$$

Here

$$\hat{y}_k(\boldsymbol{x}_i;\boldsymbol{\theta}_k) = \hat{\mu}_k + \boldsymbol{r}_k^T(\boldsymbol{x}_i - \boldsymbol{x}_1, ..., \boldsymbol{x}_i - \boldsymbol{x}_{k-1}; \hat{\boldsymbol{\theta}}_1, ..., \hat{\boldsymbol{\theta}}_{k-1}, \boldsymbol{\theta}_k)(\boldsymbol{R}_k + \boldsymbol{\Lambda}_k)^{-1}(\boldsymbol{y}_k - \hat{\mu}_k\mathbf{1}),$$

with $\hat{\mu}_k = \{\mathbf{1}^T\big(\boldsymbol{R}_k + \boldsymbol{\Lambda}_k\big)^{-1}\boldsymbol{y}_k\}/\{\mathbf{1}^T\big(\boldsymbol{R}_k + \boldsymbol{\Lambda}_k\big)^{-1}\mathbf{1}\}, \boldsymbol{\Lambda}_k = \text{diag}(\lambda_1, ..., \lambda_k)$, $\boldsymbol{y}_k = (y_1, ..., y_k)^T$. Following the adaptive nugget computation in Chapter 2,

$$\lambda_k = \frac{1}{(1-q)^3}\{\boldsymbol{r}_{k-1}^T(\boldsymbol{x}_k - \boldsymbol{x}_1, ..., \boldsymbol{x}_k - \boldsymbol{x}_{k-1}; \hat{\boldsymbol{\theta}}_1, ..., \hat{\boldsymbol{\theta}}_{k-1})(\boldsymbol{R}_{k-1} + \boldsymbol{\Lambda}_{k-1})^{-1}\boldsymbol{r}_{k-1}(\boldsymbol{x}_1 - \boldsymbol{x}_k, ..., \boldsymbol{x}_{k-1} - \boldsymbol{x}_k; \boldsymbol{\theta}_k) - q\}_+^3,$$

where $\boldsymbol{r}_{k-1}(\boldsymbol{x}_1 - \boldsymbol{x}_k, ..., \boldsymbol{x}_{k-1} - \boldsymbol{x}_k; \boldsymbol{\theta}_k)$ is an $(k-1)$-dimensional vector with ith element $R(\boldsymbol{x}_i - \boldsymbol{x}_k, \boldsymbol{\theta}_k), i = 1, ..., k-1$, and q is the algorithmic parameter usually chosen to be 0.99. Then, $\boldsymbol{\theta}$ is estimated through $\hat{\boldsymbol{\theta}}_k = \text{argmin}E(\boldsymbol{\theta}_k)$.

3.3.2 Approximate Method for Estimating Adaptive Kernel

In the current optimization scheme, given each $\boldsymbol{\theta}_k$, one needs to fit the surrogate model with adaptive kernel and minimize the residual sum of squares, which is essentially a non-linear optimization. This can actually be converted into an ordinary least squares problem through some approximation. To simplify the notation, use $\hat{y}_k(\boldsymbol{x})$ to denote $\hat{y}_k(\boldsymbol{x};\boldsymbol{\theta}_k)$. We can show that

$$\begin{aligned}\hat{y}_k(\boldsymbol{x}) =& \hat{y}_{k-1}(\boldsymbol{x}) - e_i\{\boldsymbol{r}_{k-1}^T(\boldsymbol{x}_i - \boldsymbol{x}, ..., \boldsymbol{x} - \boldsymbol{x}_{k-1}; \hat{\boldsymbol{\theta}}_1, ..., \hat{\boldsymbol{\theta}}_{k-1})(\boldsymbol{R}_{k-1} + \boldsymbol{\Lambda}_{k-1})^{-1} \\ & \boldsymbol{r}_{k-1}(\boldsymbol{x}_1 - \boldsymbol{x}_k, ..., \boldsymbol{x}_{k-1} - \boldsymbol{x}_k; \boldsymbol{\theta}_k) - R(\boldsymbol{x} - \boldsymbol{x}_k; \boldsymbol{\theta}_k)\}/\{1 + \lambda_k - \\ & \boldsymbol{r}_{k-1}^T(\boldsymbol{x}_k - \boldsymbol{x}_1, ..., \boldsymbol{x}_k - \boldsymbol{x}_{k-1}; \hat{\boldsymbol{\theta}}_1, ..., \hat{\boldsymbol{\theta}}_{k-1})(\boldsymbol{R}_{k-1} + \boldsymbol{\Lambda}_{k-1})^{-1}\boldsymbol{r}_{k-1}(\boldsymbol{x}_1 - \boldsymbol{x}_k, ..., \boldsymbol{x}_{k-1} - \boldsymbol{x}_k; \boldsymbol{\theta}_k)\},\end{aligned} \tag{3.10}$$

where $e_i = y_i - \hat{y}_{k-1}(\boldsymbol{x}_i)$. Thus, a reasonable approximation of (3.10) is $\hat{y}_k(\boldsymbol{x}) \approx \hat{y}_{k-1}(\boldsymbol{x}) + cR(\boldsymbol{x} - \boldsymbol{x}_k; \boldsymbol{\theta}_k)$ with c being constant so that

$$E(\boldsymbol{\theta}_k) = \sum_{i \in \mathcal{N}_k}(y_i - \hat{y}_k(\boldsymbol{x}_i;\boldsymbol{\theta}_k))^2 \approx \sum_{i \in \mathcal{N}_k}(e_i - cR(\boldsymbol{x}_i - \boldsymbol{x}_k;\boldsymbol{\theta}_k))^2. \tag{3.11}$$

Since the residuals can have opposite signs and $\boldsymbol{x}_k$ is the point with the largest residual, it is better for us to minimize $\tilde{E}(\boldsymbol{\theta}_k) = \sum_{i \in \mathcal{N}_k}(e_i - e_k + c\{1 - R(\boldsymbol{x}_i - \boldsymbol{x}_k;\boldsymbol{\theta}_k)\})^2$.

Assuming the errors are uniformly distributed, we can use $\hat{c}_k = \frac{4p+2}{4p+1}\{\max_{i\in\mathcal{N}_k}(e_i) - \max_{i\in\mathcal{N}_k}(e_i)\}\text{sign}(e_k)$ to estimate c. Then the original nonlinear optimization problem of minimizing $E(\boldsymbol{\theta}_k)$ can be converted to

$$\min \sum_{i\in\mathcal{N}_k}\{-\log(\frac{e_i - e_k}{\hat{c}_k} + 1) - \sum_{j=1}^{p}(x_{ij} - x_{kj})^2\theta_{kj}\}^2, \tag{3.12}$$

which can be easily solved using ordinary least squares. Note that the time complexity of estimating all the $\boldsymbol{\theta}_k, k = 1, ..., n$ is $\mathcal{O}(np^3)$ while that of one likelihood evaluation in ordinary kriging is $\mathcal{O}(n^3)$. The ease of computation empowers the method to handle large datasets.

3.3.3 Bayesian Approach to Stabilize Kernel Estimates

Since the estimates of the kernel parameters are based on neighborhoods, they are subject to larger variability. We can overcome this by adopting the idea of Bayesian regression. Assume the prior is $\boldsymbol{\theta}_k \sim N(\boldsymbol{\theta}_0, \boldsymbol{\Sigma}_0)$, and we have the linear regression model $\boldsymbol{z}^{(k)} \sim N(\boldsymbol{W}^{(k)}\boldsymbol{\theta}_k, \sigma^2\boldsymbol{I})$ based on (3.12), where $z_i^{(k)} = -\log(\frac{e_i - e_k}{\hat{c}_k} + 1), w_{ij}^{(k)} = (x_{ij} - x_{kj})^2, i \in \mathcal{N}_k, j = 1, ..., p, k = 1, ..., n$, and σ^2 is the variance for the error. Thus $\boldsymbol{z}^{(k)}|\boldsymbol{\theta}_0, \boldsymbol{\Sigma}_0 \sim N(\boldsymbol{W}^{(k)}\boldsymbol{\theta}_0, \sigma^2\boldsymbol{I} + \boldsymbol{W}^{(k)}\boldsymbol{\Sigma}_0\boldsymbol{W}^{(k)T})$. Let $\boldsymbol{\Sigma}_k = \sigma^2\boldsymbol{I} + \boldsymbol{W}^{(k)}\boldsymbol{\Sigma}_0\boldsymbol{W}^{(k)T}$ and the Empirical Bayes estimator for $\boldsymbol{\theta}_0$ is given by

$$\hat{\boldsymbol{\theta}}_0 = (\sum_{i=1}^{k}\boldsymbol{W}^{(k)T}\boldsymbol{\Sigma}_k^{-1}\boldsymbol{W}^{(k)})^{-1}(\sum_{i=1}^{k}\boldsymbol{W}^{(k)T}\boldsymbol{\Sigma}_k^{-1}\boldsymbol{z}^{(k)}).$$

To simplify the computations, we adopt the g-prior for $\boldsymbol{\theta}_k$ such that $\boldsymbol{\Sigma}_0 = g(\boldsymbol{W}^{(k)T}\boldsymbol{W}^{(k)})^{-1}$. Let $\boldsymbol{W} = \left[\boldsymbol{W}^{(1)T}, \boldsymbol{W}^{(2)T}..., \boldsymbol{W}^{(k)T}\right]^T$ and $\boldsymbol{z} = \left[\boldsymbol{z}^{(1)T}, \boldsymbol{z}^{(2)T}..., \boldsymbol{z}^{(k)T}\right]^T$. Then the posterior of $\boldsymbol{\theta}^{(k)}$ given data can be expressed by

$$\boldsymbol{\theta}_k|\boldsymbol{W}, \boldsymbol{z} \sim N((\frac{1}{\sigma^2} + \frac{1}{g})^{-1}(\frac{\tilde{\boldsymbol{\theta}}_k}{\sigma^2} + \frac{\hat{\boldsymbol{\theta}}_0}{g}), (\frac{1}{\sigma^2} + \frac{1}{g})^{-1}(\boldsymbol{W}^{(k)T}\boldsymbol{W}^{(k)})^{-1}),$$

where $\tilde{\boldsymbol{\theta}}_k = (\boldsymbol{W}^{(k)T}\boldsymbol{W}^{(k)})^{-1}\boldsymbol{W}^{(k)T}\boldsymbol{z}^{(k)}$. Since it is hard to obtain an estimator for g/σ^2, let $\eta = \frac{g}{g+\sigma^2}$ be an algorithmic parameter and $\boldsymbol{\theta}_k$ is estimated through $\hat{\boldsymbol{\theta}}_k = \eta\tilde{\boldsymbol{\theta}}_k + (1-\eta)\hat{\boldsymbol{\theta}}_0$. After estimating all the correlation parameters $\hat{\boldsymbol{\theta}}_k, k = 1, ..., n$, they are plugged into (3.8) to build our final adaptive kernel predictor. It can be shown that automatic kriging is further enhanced by the capability of having adaptive kernel. The AK predictor is also compared with ordinary kriging implemented via R package *GPfit*. Suppose the truth is the 1d function used earlier and we generate $n = 20$ design points $\boldsymbol{X}$ from uniform distribution $U(0, 1)$. The testing data is $n_{test} = 1001$ equally spaced points in $[0, 1]$. We repeat the simulations for $N = 100$ times and summarize the MSPE for both AK and GPfit. According to Figure 3.10, adaptive kernel achieves better performance than ordinary kriging with uniformly distributed design points, and it also performs well for non-uniform design as shown in Figure 3.11.

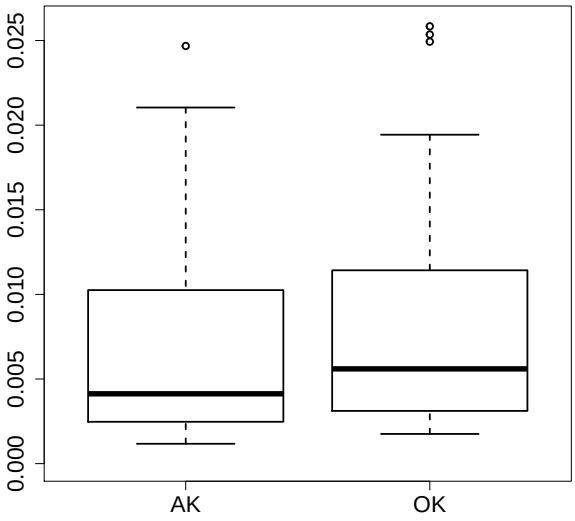

Figure 3.10: The comparison of adaptive kernel predictor and ordinary kriging with 20 design points from $U(0, 1)$ using 1d function

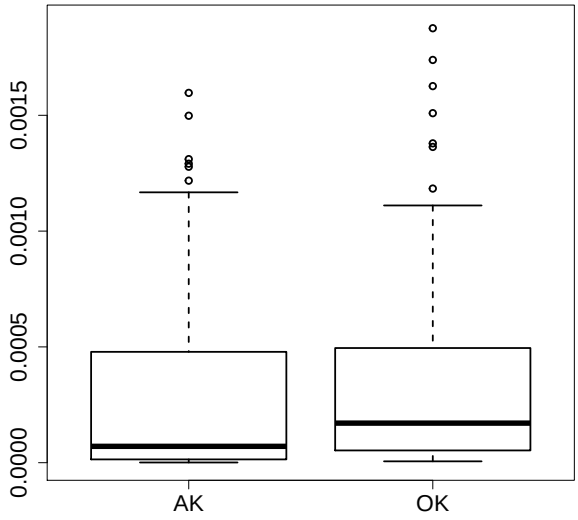

Figure 3.11: The comparison of adaptive kernel predictor and ordinary kriging with 70 points from non-uniform design using 1d function: 56 points from U(0,0.5),14 points from U(0.5,1)

We also use the Franke 2d function

$$\begin{aligned} y =& 0.75\exp\{-\frac{(9x_1-2)^2}{4}-\frac{(9x_2-2)^2}{4}\}+0.75\exp\{-\frac{(9x_1+1)^2}{49}-\frac{9x_2+1}{10}\} \\ &+0.5\exp\{-\frac{(9x_1-7)^2}{4}-\frac{(9x_2-3)^2}{4}\}-0.2\exp\{-(9x_1-4)^2-(9x_2-7)^2\} \end{aligned}$$

as the true function with design size $n = 30$, among which 22 points are generated from $U(0, 0.5)^2$ and 8 points are generated from $U(0, 1)^2$. The testing data is a $n_{test} = 1000$ Sobol sequence. Again, we run the simulation 100 times and summarize the MSPE for adaptive kernel and GPfit. It can be seen from Figure 3.12 and that AK still has better overall performance than OK. We can show similar results if the ordinary kriging is implemented through the *RobustGaSP* R library [75].

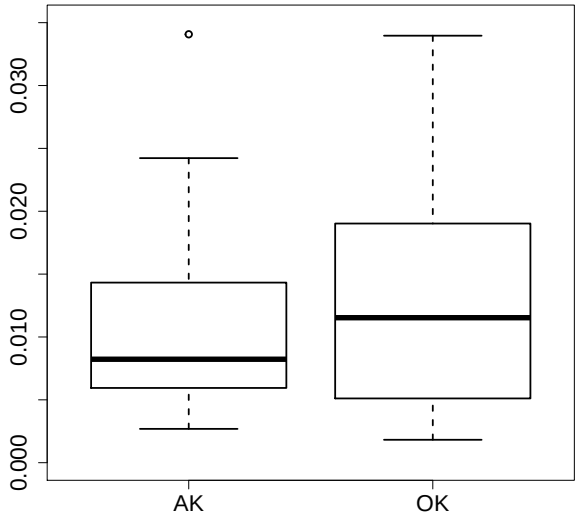

Figure 3.12: The comparison of adaptive nugget predictor and ordinary kriging with 22 design points from $U[0, 0.5]^2$ and 8 design points from $U[0, 1]^2$ using Franke 2d function

Since the correlation matrix for adaptive kernel predictor is asymmetric, it poses challenges for us to build a confidence interval for the predictor. A heuristic mean squared prediction error (MSPE) can be computed as

$$v(\boldsymbol{x}) = \text{MSPE}\{\hat{y}(\boldsymbol{x})\} = \hat{\tau}^2\{1 - \tilde{\boldsymbol{r}}^T(\boldsymbol{x})(\tilde{\boldsymbol{R}} + \boldsymbol{\Lambda})^{-1}\tilde{\boldsymbol{r}}(\boldsymbol{x})\},$$

where $\tilde{\boldsymbol{r}}(\boldsymbol{x}) = \{\boldsymbol{r}_n^T(\boldsymbol{x}-\boldsymbol{x}_1, ..., \boldsymbol{x}-\boldsymbol{x}_n; \hat{\boldsymbol{\theta}}_1, ..., \hat{\boldsymbol{\theta}}_n)+\boldsymbol{r}_n^T(\boldsymbol{x}_1-\boldsymbol{x}, ..., \boldsymbol{x}_n-\boldsymbol{x}; \hat{\boldsymbol{\theta}}_x)\}/2$, $\boldsymbol{r}_n(\boldsymbol{x}_1 - \boldsymbol{x}, ..., \boldsymbol{x}_n - \boldsymbol{x}; \hat{\boldsymbol{\theta}}_x)$ is an n-dimensional column vector with ith element $R(\boldsymbol{x}_i - \boldsymbol{x}; \hat{\boldsymbol{\theta}}_x)$, $\tilde{\boldsymbol{R}} = (\boldsymbol{R}+\boldsymbol{R}^T)/2$. $\hat{\boldsymbol{\theta}}_x$ can be estimated using least squares similar to (3.9). An approximate $(1 - \alpha)\%$ confidence interval for $y(\boldsymbol{x})$ can be constructed as

$$\left[\hat{y}(\boldsymbol{x}) - z_{1-\alpha/2}\sqrt{v(\boldsymbol{x})}, \hat{y}(\boldsymbol{x}) + z_{1-\alpha/2}\sqrt{v(\boldsymbol{x})}\right].$$

3.4 Conclusions

This chapter proposes innovative ideas to enhance the automatic kriging when more computational resources are available. The two-stage version of adaptive nugget can address the commonly-seen instability phenomenon in Gaussian process model with large design size or close points. In addition, the two-stage predictor demonstrates its advantage over the widely-used R package in various examples with different designs and dimensions. Due to the sequential nature of the adaptive nugget predictor, we have the flexibility to balance the estimation accuracy and computational complexity. For the adaptive kernel approach, the estimation is improved by making use of the response data. After applying approximation to the nonlinear optimization problem, adaptive kernel estimates can be obtained without much additional computation cost.

REFERENCES

[1] M. C. Kennedy and A. O'Hagan, "Bayesian calibration of computer models," *Journal of the Royal Statistical Society. Series B, Statistical Methodology*, vol. 63, no. 3, pp. 425–464, 2001.

[2] M. J. Bayarri, J. O. Berger, R. Paulo, J. Sacks, J. A. Cafeo, J. Cavendish, C.-H. Lin, and J. Tu, "A framework for validation of computer models," *Technometrics*, vol. 49, no. 2, pp. 138–154, 2007.

[3] D. Higdon, J. Gattiker, B. Williams, and M. Rightley, "Computer model calibration using high-dimensional output," *Journal of the American Statistical Association*, vol. 103, no. 482, pp. 570–583, 2008.

[4] S. Wang, W. Chen, and K.-L. Tsui, "Bayesian validation of computer models," *Technometrics*, vol. 51, no. 4, pp. 439–451, 2009.

[5] J. Goh, D. Bingham, J. P. Holloway, M. J. Grosskopf, C. C. Kuranz, and E. Rutter, "Prediction and computer model calibration using outputs from multifidelity simulators," *Technometrics*, vol. 55, no. 4, pp. 501–512, 2013.

[6] R. Tuo and C. F. J. Wu, "A theoretical framework for calibration in computer models: parameterization, estimation and convergence properties," *The Annals of Statistics*, vol. 43, no. 6, pp. 2331–2352, 2015.

[7] V. R. Joseph and H. Yan, "Engineering-driven statistical adjustment and calibration," *Technometrics*, vol. 57, no. 2, pp. 257–267, 2015.

[8] Z. Jiang, D. W. Apley, and W. Chen, "Surrogate preposterior analyses for predicting and enhancing identifiability in model calibration," *International Journal for Uncertainty Quantification*, vol. 5, no. 4, pp. 341–359, 2015.

[9] M. Pratola and D. Higdon, "Bayesian additive regression tree calibration of complex high-dimensional computer models," *Technometrics*, vol. 58, no. 2, pp. 166–179, 2016.

[10] J. P. Klein and M. L. Moeschberger, *Survival analysis: techniques for censored and truncated data*. Springer, New York, 2003.

[11] J. D. Kalbfleisch and R. L. Prentice, *The statistical analysis of failure time data*. John Wiley & Sons, Hoboken, NJ, 2011, vol. 360.

[12] W. Q. Meeker and L. A. Escobar, *Statistical methods for reliability data*. John Wiley & Sons, New York, 2014.

[13] O. Dubrule and C. Kostov, "An interpolation method taking into account inequality constraints: i. methodology," *Mathematical geology*, vol. 18, no. 1, pp. 33–51, 1986.

[14] A. Journel, "Constrained interpolation and qualitative informationthe soft kriging approach," *Mathematical Geology*, vol. 18, no. 3, pp. 269–286, 1986.

[15] X. Wang and J. O. Berger, "Estimating shape constrained functions using gaussian processes," *SIAM/ASA Journal on Uncertainty Quantification*, vol. 4, no. 1, pp. 1–25, 2016.

[16] S. Manley, J. Skotheim, L. Mahadevan, and D Weitz, "Gravitational collapse of colloidal gels," *Physical Review Letters*, vol. 94, no. 21, pp. 3–6, 2005.

[17] P. Groot and P. Lucas, "Gaussian process regression with censored data using expectation propagation," in *Sixth European Workshop on Probabilistic Graphical Models*, Granada, Spain, 2012.

[18] T. Amemiya, "Tobit models: a survey," *Journal of econometrics*, vol. 24, no. 1-2, pp. 3–61, 1984.

[19] T. P. Minka, "Expectation propagation for approximate bayesian inference," in *Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence*, Morgan Kaufmann Publishers Inc. San Francisco, CA, 2001, pp. 362–369.

[20] A. Genz, "Numerical computation of multivariate normal probabilities," *Journal of computational and graphical statistics*, vol. 1, no. 2, pp. 141–149, 1992.

[21] A. Genz, F. Bretz, T. Miwa, X. Mi, F. Leisch, F. Scheipl, and T. Hothorn, "Mvtnorm: multivariate normal and t distributions," *R package version 3.2.5, URL http://mvtnorm.R-forge.R-project.org*, 2016.

[22] Z. I. Botev, "The normal law under linear restrictions: simulation and estimation via minimax tilting," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 2016.

[23] S. Banerjee, A. E. Gelfand, A. O. Finley, and H. Sang, "Gaussian predictive process models for large spatial data sets," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 70, no. 4, pp. 825–848, 2008.

[24] G. Tallis, "The moment generating function of the truncated multi-normal distribution," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 23, no. 1, pp. 223–229, 1961.

[25] W. C. Horrace, "Some results on the multivariate truncated normal distribution," *Journal of Multivariate Analysis*, vol. 94, no. 1, pp. 209–221, 2005.

[26] R. Kan and C. Robotti, "On moments of folded and truncated multivariate normal distributions," *Available at SSRN*, 2016.

[27] V. R. Joseph, "Bayesian computation using design of experiments-based interpolation technique (with discussions)," *Technometrics*, vol. 54, no. 3, pp. 209–225, 2012.

[28] M. Plumlee, "Bayesian calibration of inexact computer models," *Journal of the American Statistical Association*, no. just-accepted, 2016.

[29] C.-J. Chang and V. R. Joseph, "Model calibration through minimal adjustments," *Technometrics*, vol. 56, no. 4, pp. 474–482, 2014.

[30] G. Matheron, "Principles of geostatistics," *Economic geology*, vol. 58, no. 8, pp. 1246–1266, 1963.

[31] Y. Li, S. H. Ng, M. Xie, and T. Goh, "A systematic comparison of metamodeling techniques for simulation optimization in decision support systems," *Applied Soft Computing*, vol. 10, no. 4, pp. 1257–1273, 2010.

[32] M. L. Stein, Z. Chi, and L. J. Welty, "Approximating likelihoods for large spatial data sets," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 66, no. 2, pp. 275–296, 2004.

[33] T. Gneiting, "Compactly supported correlation functions," *Journal of Multivariate Analysis*, vol. 83, no. 2, pp. 493–508, 2002.

[34] R. Furrer, M. G. Genton, and D. Nychka, "Covariance tapering for interpolation of large spatial datasets," *Journal of Computational and Graphical Statistics*, vol. 15, no. 3, pp. 502–523, 2006.

[35] N. Cressie and G. Johannesson, "Fixed rank kriging for very large spatial data sets," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 70, no. 1, pp. 209–226, 2008.

[36] C. G. Kaufman, M. J. Schervish, and D. W. Nychka, "Covariance tapering for likelihood-based estimation in large spatial data sets," *Journal of the American Statistical Association*, vol. 103, no. 484, pp. 1545–1555, 2008.

[37] H. Sang and J. Z. Huang, "A full scale approximation of covariance functions for large spatial data sets," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 74, no. 1, pp. 111–132, 2012.

[38] W. Kleiber and D. W. Nychka, "Equivalent kriging," *Spatial Statistics*, vol. 12, pp. 31–49, 2015.

[39] R. B. Gramacy and D. W. Apley, "Local gaussian process approximation for large computer experiments," *Journal of Computational and Graphical Statistics*, vol. 24, no. 2, pp. 561–578, 2015.

[40] A. Datta, S. Banerjee, A. O. Finley, and A. E. Gelfand, "Hierarchical nearest-neighbor gaussian process models for large geostatistical datasets," *Journal of the American Statistical Association*, vol. 111, no. 514, pp. 800–812, 2016.

[41] R. B. Gramacy and B. Haaland, "Speeding up neighborhood search in local gaussian process prediction," *Technometrics*, vol. 58, no. 3, pp. 294–303, 2016.

[42] C. Park and D. Apley, "Patchwork kriging for large-scale gaussian process regression," *arXiv preprint arXiv:1701.06655*, 2017.

[43] R. B. Gramacy and H. K. Lee, "Adaptive design and analysis of supercomputer experiments," *Technometrics*, vol. 51, no. 2, pp. 130–145, 2009.

[44] B. Haaland, P. Z. Qian, *et al.*, "Accurate emulators for large-scale computer experiments," *The Annals of Statistics*, vol. 39, no. 6, pp. 2974–3002, 2011.

[45] S. Mak, C.-L. Sung, X. Wang, S.-T. Yeh, Y.-H. Chang, V. R. Joseph, V. Yang, and C. Wu, "An efficient surrogate model of large eddy simulations for design evaluation and physics extraction," *arXiv preprint arXiv:1611.07911*, 2016.

[46] C. J. Paciorek, B. Lipshitz, W. Zhuo, C. G. Kaufman, R. C. Thomas, *et al.*, "Parallelizing gaussian process calculations in r," *arXiv preprint arXiv:1305.4886*, 2013.

[47] V. R. Joseph, "Limit kriging," *Technometrics*, vol. 48, no. 4, pp. 458–466, 2006.

[48] M. R. Lee and A. B. Owen, "Single nugget kriging," *arXiv preprint arXiv:1507.05128*, 2015.

[49] D. Shepard, "A two-dimensional interpolation function for irregularly-spaced data," in *Proceedings of the 1968 23rd ACM national conference*, ACM, 1968, pp. 517–524.

[50] V. R. Joseph and L. Kang, "Regression-based inverse distance weighting with applications to computer experiments," *Technometrics*, vol. 53, no. 3, pp. 254–265, 2011.

[51] R. B. Gramacy and H. K. Lee, "Cases for the nugget in modeling computer experiments," *Statistics and Computing*, vol. 22, no. 3, pp. 713–722, 2012.

[52] C.-Y. Peng and C. J. Wu, "On the choice of nugget in kriging modeling for deterministic computer experiments," *Journal of Computational and Graphical Statistics*, vol. 23, no. 1, pp. 151–168, 2014.

[53] P. Ranjan, R. Haynes, and R. Karsten, "A computationally stable approach to gaussian process interpolation of deterministic computer simulation data," *Technometrics*, vol. 53, no. 4, pp. 366–378, 2011.

[54] P. W. Goldberg, C. K. Williams, and C. M. Bishop, "Regression with input-dependent noise: a gaussian process treatment," in *Advances in neural information processing systems*, 1998, pp. 493–499.

[55] J Yin, S. Ng, and K. Ng, "Kriging model with modified nugget effect for random simulation with heterogeneous variances," in *Industrial Engineering and Engineering Management, 2008. IEEM 2008. IEEE International Conference on*, IEEE, 2008, pp. 1714–1718.

[56] B. Ankenman, B. L. Nelson, and J. Staum, "Stochastic kriging for simulation metamodeling," *Operations research*, vol. 58, no. 2, pp. 371–382, 2010.

[57] J. P. Kleijnen and W. C. Van Beers, "Robustness of kriging when interpolating in random simulation with heterogeneous variances: some experiments," *European Journal of Operational Research*, vol. 165, no. 3, pp. 826–834, 2005.

[58] D. Cervone and N. S. Pillai, "Gaussian process regression with location errors," *arXiv preprint arXiv:1506.08256*, 2015.

[59] M. Binois, R. B. Gramacy, and M. Ludkovski, "Practical heteroskedastic gaussian process modeling for large simulation experiments," *arXiv preprint arXiv:1611.05902*, 2016.

[60] T. C. Haas, "Kriging and automated variogram modeling within a moving window," *Atmospheric Environment. Part A. General Topics*, vol. 24, no. 7, pp. 1759–1769, 1990.

[61] M. N. Gibbs, "Bayesian gaussian processes for regression and classification," PhD thesis, University of Cambridge Cambridge, England, 1998.

[62] A. M. Schmidt and A. O'Hagan, "Bayesian inference for non-stationary spatial covariance structure via spatial deformations," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 65, no. 3, pp. 743–758, 2003.

[63] M. Plumlee and D. W. Apley, "Lifted brownian kriging models," *Technometrics*, vol. 59, no. 2, pp. 165–177, 2017.

[64] G. Fasshauer and M. McCourt, *Kernel-based approximation methods using Matlab*. World Scientific Publishing Co Inc, 2015, vol. 19.

[65] P. Abrahamsen, *A review of Gaussian random fields and correlation functions*. Norsk Regnesentral/Norwegian Computing Center, 1997.

[66] Y. Xiong, W. Chen, D. Apley, and X. Ding, "A non-stationary covariance-based kriging method for metamodelling in engineering design," *International Journal for Numerical Methods in Engineering*, vol. 71, no. 6, pp. 733–756, 2007.

[67] M. S. Floater and A. Iske, "Multistep scattered data interpolation using compactly supported radial basis functions," *Journal of Computational and Applied Mathematics*, vol. 73, no. 1-2, pp. 65–78, 1996.

[68] L. Kang and V. R. Joseph, "Kernel approximation: from regression to interpolation," *SIAM/ASA Journal on Uncertainty Quantification*, vol. 4, no. 1, pp. 112–129, 2016.

[69] S. Ba and V. R. Joseph, "Composite gaussian process models for emulating expensive functions," *The Annals of Applied Statistics*, vol. 6, no. 4, pp. 1838–1860, 2012.

[70] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient global optimization of expensive black-box functions," *Journal of Global optimization*, vol. 13, no. 4, pp. 455–492, 1998.

[71] P. Ranjan, D. Bingham, and G. Michailidis, "Sequential experiment design for contour estimation from complex computer codes," *Technometrics*, vol. 50, no. 4, pp. 527–541, 2008.

[72] V. Picheny, "Multiobjective optimization using gaussian process emulators via stepwise uncertainty reduction," *Statistics and Computing*, vol. 25, no. 6, pp. 1265–1280, 2015.

[73] I. Pesenson, Q. T. Le Gia, A. Mayeli, H. Mhaskar, and D.-X. Zhou, *Frames and Other Bases in Abstract and Function Spaces*. Springer, 2017, pp. 917–942.

[74] C. Q. Lam, "Sequential adaptive designs in computer experiments for response surface model fit," PhD thesis, The Ohio State University, 2008.

[75] M. Gu, "Robust uncertainty quantification and scalable computation for computer models with massive output," PhD thesis, Duke University, 2016.